Trust Graphs for Belief Revision: Framework and Implementation

Aaron Hunter[1,*], Sam Tadey[1]

[1]British Columbia Institute of Technology
Burnaby, Canada

Abstract

Trust plays a role in the process of belief revision. When information is reported by another agent, it should only be believed if the reporting agent is trusted as an authority over some relevant domain. In practice, an agent will be trusted on a particular topic if they have provided accurate information on that topic in the past. In this paper, we demonstrate how an agent can construct a model of knowledge-based trust based on the accuracy of past reports. We then show how this model of trust can be used in conjunction with Ordinal Conditional Functions to define two approaches to trust-influenced belief revision. In the first approach, strength of trust and strength of belief are assumed to be incomparable as they are on different scales. In the second approach, they are aggregated in a natural manner. We then describe a software tool for modelling and reasoning about trust and belief change. Our software allows a trust graph to be updated incrementally by looking at the accuracy of past reports. After constructing a trust graph, the software can then compute the result of AGM-style belief revision using two different approaches to incorporating trust.

Keywords

belief revision, knowledge representation, trust

1. Introduction

Belief revision is concerned with the manner in which an agent incorporates new information that may be inconsistent with their current beliefs. In general, the belief revision literature assumes that new information is more reliable than the initial beliefs; in this case, new information must always be believed following belief revision. However, in many practical situations this is not a reasonable assumption. In practice, we need to take into account the extent to which the source of the new information is *trusted*. In this paper, we demonstrate how an agent can actually build trust in a source, based on past reports.[1]

Suppose that an agent believes ϕ to be true, and they are being told by an agent R that ϕ is not true. In this kind of situation, we will use *ranking functions* to represent both the initial strength of belief in ϕ as well as the level of trust in R. Significantly, however, the trust in R is not uniform over all formulas. Each information source is trusted to different degrees on different topics. The extent to which R is trusted on a particular topic is determined by how frequently they have made accurate reports on that topic in the past.

In the rest of the paper, we proceed as follows. In the next section, we give a motivating example that will be used throughout the paper. We then review formal preliminaries related to belief revision and trust. We then introduce *trust graphs*, our formal model of trust. We define a simple approach for building a trust graph from past revisions, and prove some basic results. We then demonstrate how trust rankings can influence belief revision in two different ways. First, we consider the naive case, where the strength of trust is independent of the strength of belief. Second, we consider the more complex case, where strength of trust is aggregated with strength of belief.

Finally, we describe and implemented software tool that automates this entire process. The software presented here is a useful addition to the relatively small collection existing belief revision solvers, because it extends the class of practical problems that we can model and solve. To the best of our knowledge, the software presented in this paper is the first implemented system that incrementally builds a model of trust that is specifically intended to inform the process of belief revision.

2. Preliminaries

2.1. Motivating Example

Consider a situation where there are two rooms A and B located inside a building. There are two agents, which we call *Absent* and *Present*. Informally, *Absent* is not in the building whereas *Present* is in the building. These agents

NMR 2022: 20th International Workshop on Non-Monotonic Reasoning, August 07–09, 2022, Haifa, Israel
*Corresponding author.
aaronhunter@bcit.ca (A. Hunter); samtadey10@gmail.com (S. Tadey)

[1]The content of sections 1-4 of this paper have previously been published in [1]. The implementation discussed in section 5 is new, and was not discussed in that paper.

communicate about the status of the lights in each room. For simplicity, we say that A is true when the light is on in room A and we say B is true when the light is on in room B.

We focus on the beliefs of *Absent*, who initially thinks that the light in room A is on and the light in room B is off. Now suppose that *Present* sends a message that asserts the light is off in A and the light is on in room B. If *Present* is completely trusted, this is not a problem; the report simply leads *Absent* to believe they were incorrect about the lights.

But suppose that *Present* has given incorrect reports in the past. We can collect these reports, and check to see when they have been correct and when they have been incorrect. For example, suppose that *Present* is always correct about the light status in room A, whereas they are often incorrect about the light status in room B. We might draw the conclusion that they are normally physically in the room A, and that they are too lazy to walk to a another room to check the lights.

Formally, *Absent* does not need a plausible story to explain the mistakes in the reports; they need some mechanism for modelling trust over different propositions. By looking at the accuracy of reports on different topics, they build a model of trust that allows information reported from *Present* to be incorporated appropriately. In this paper, we develop formal machinery that is suitable for capturing all facets of this seemingly simple example.

2.2. Belief Revision

We assume an underlying set $\mathbf{V}$ of propositional variables. A *formula* is a propositional combination of elements of $\mathbf{V}$, using the usual connectives $\rightarrow, \wedge, \vee, \neg$. We will assume that $\mathbf{V}$ is finite in this paper, though that need not be the case in general. A *state* is a propositional interpretation over $\mathbf{V}$, which assigns boolean values to all variables. We will normally specify a state by giving the set of variables that are true. A *belief state* is a set of states, informally representing the set of states that an agent considers possible. We let $|\phi|$ denote the set of states where the formula ϕ is true.

The dominant approach to belief revision is the AGM approach. A revision operator is a function $*$ that maps a belief state K and a formula ϕ to a new belief state $K * \phi$. An AGM revision operator is a revision operator that satisfies the so-called AGM postulates. We refer the reader to [2] for a complete introduction to the AGM theory of belief revision.

Although we are concerned with AGM revision at times, in this paper we actually define the beliefs of an agent in terms of Ordinal Conditional Functions (OCFs) [3], which are also called *ranking functions*. An OCF is a function κ that maps every state s to an ordinal $\kappa(s)$. Informally, if $\kappa(s_1) < \kappa(s_2)$, this is understood to mean that the agent considers it more likely that s_1 is the actual state, as opposed to s_2. Note that κ defines a belief state $Bel(\kappa)$ as follows:

$$Bel(\kappa) = \{s \mid \kappa(s) \text{ is minimal }\}.$$

We can also define a revision operator $*$ associated with κ as follows:

$$Bel(\kappa) * \phi = \min_{\kappa}(|\phi|).$$

The operator on belief states specified in this manner defines an AGM belief revision operator, for any underlying OCF.

2.3. Trust

The notion of trust plays an important role in many applications, including security [4, 5] and multi-agent systems [6, 7]. In this paper, we are concerned primarily with *knowledge-based* trust. That is, we are concerned with the extent to which one agent trusts another to have the knowledge required to be trusted on particular statements. This is distinct from trust related to honesty or deception.

We refer occasionally to *trust-senstive* belief revision operators [8]. Trust-sensitive belief revision operators are defined with respect to a trust-partition over states. The equivalence classes of a trust partition Π consist of states that can not be distinguished by a particular reporting agent. In our motivating example, we might define a trust partition for *Present* that consists of two equivalence classes: one that includes all states where the light is on in room A, and one that includes all states where the light is off in room A. In this case, *Present* is informally trusted to be able to tell if the light in room A is on or off. However, *Present* is not trusted to be able to tell if the light in room B is on or off.

A trust-sensitive revision operator $*_\Pi$ is defined with respect to a given AGM revision operator $*$ and a trust partition Π. The operator $*_\Pi$ operates in two steps when an agent is given a report ϕ. First, we find the set $\Pi(\phi)$ of all states that are related by Π to a model of ϕ. Then we perform regular AGM revision with this expanded set of states as input. Hence, the model of trust is essentially used to pre-process the formula for revision, by expanding it to ignore distinctions that we do not trust the reporter to be able to make.

2.4. Trust Rankings

We can also define trust in terms of a distance function between states. The notion of distance required is generally an ultrametric.

Definition 1. *An* ultrametric *is a binary function d over a set X, such that for all $x, y, z \in X$:*

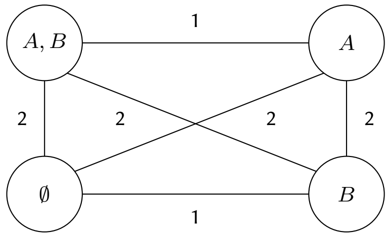

Figure 1: A Trust Graph

- $d(x, y) \geq 0$.
- $d(x, y) = 0$ *if and only if* $x = y$.
- $d(x, y) = d(y, x)$.
- $d(x, z) \leq \max\{d(x, y), d(y, z)\}$.

If we remove condition 2*, then* d *is a pseudo-ultrametric.*

The following definition of a trust ranking is given in [9].

Definition 2. *For any propositional vocabulary, a* trust ranking *is a pseudo-ultrametric over the set* S *of all states.*

A trust ranking is intended to capture the degree to which an agent is trusted to distinguish between states in a graph. If $d(s_1, s_2)$ is large, this means the agent can be trusted to distinguish the states s_1 and s_2. However, if the distance is small, they can not be trusted to draw this distinction.

3. Building Trust

3.1. Trust Graphs

We now turn to our main problem: building a notion of trust from data. We assume throughout this paper a fixed, finite vocabulary $\mathbf{V}$. All states, belief states, and formulas will be defined with respect to this underlying vocabulary.

Definition 3. *Let* S *be the set of states over* $\mathbf{V}$*. A* trust graph *over* S *is a pair* $\langle S, w \rangle$*, where* $w : S \times S \to \mathbf{N}$*.*

Hence, a trust graph is just a complete weighted graph along with a distance between states. Informally, a trust graph represents the trust held in another agent. The weight on the edge between two states s_1 and s_2 is an indication of how strongly the agent is trusted to *directly* distinguish between those states.

Example 1. *Consider the motivating example, in the case where* Absent *trusts* Present *more strongly to check if the light in room* A *is on as opposed to room* B*. This could be captured by the trust graph in Figure 1, by having a higher weight on edges that connect states that differ on the value of* A*. Note that the minimax distance* d *can easily be calculated from this graph.*

The edge weights represent how strongly a reporting agent is trusted to distinguish between a pair of states. If the weight is high, we interpret this to mean that the agent is strongly trusted to distinguish between the states. If the weight is low, then the reporting agent is not trusted to distinguish the states.

In order to build a notion of trust in an agent, we need to have a history of the past reports that agent has provided. Our basic approach is to assume that we start with a set of statements that a reporting agent has made in the past, along with an indication of whether the reports were correct or not.

Definition 4. *A* report *is a pair* (ϕ, i)*, where* ϕ *is a formula and* $i \in \{0, 1\}$*. A* report history *is a multi-set of reports.*

We let Φ, possibly with subscripts, range over report histories. A report history Φ represents all of the claims that an agent has truthfully or falsely claimed in the past. Informally, if $(\phi, 1) \in \Phi$ then the agent in question has reported ϕ in the past in a situation where ϕ was shown to be true. On the other hand, $(\phi, 0) \in \Phi$ means that ϕ has been reported in a situation where it was false.

3.2. Construction from Reports

Suppose we start with a trust graph in which the reporting agent is essentially trusted to be able to distinguish all states, with a default confidence level. For each true report in the history, we strengthen our trust in the reporting agent's ability to distinguish certain states. For each false report, we weaken our trust.

Definition 5. *For any* $n > 0$*, the* initial trust graph $T_n = \langle S, w \rangle$ *where* S *is the set of states, and* w *is defined such that* $w(s, t) = 0$ *if* $s = t$ *and* $w(s, t) = n$ *otherwise.*

The idea of the initial trust graph is that the reporting agent is trusted to distinguish between all states equally well.

We are now interested in giving a procedure that takes a report history, and returns a trust graph; this is presented in Algorithm 1. The algorithm looks at each report in the history R, and it increases the weight on edges where there have been true reports and decreases the weight on edges where there have been false reports.

Proposition 1. *Given a report history* R*, the weighted graph returned by Algorithm 1 is a trust graph.*

This result relies on the fact that w only returns non-negative values; this is guaranteed by the choice of n for the initial trust graph.

Example 2. *We return to our running example. Suppose that we have no initial assumptions about the trust held*

Algorithm 1 Construct_from(R)

Input R, a report history.
$n \leftarrow$ size of R.
$T = \langle S, w \rangle$ is the initial trust graph for n.
while $R \neq \emptyset$ **do**
  Get some $(\phi, i) \in R$
  if $i = 0$ **then**
    $w(s_1, s_2) \leftarrow w(s_1, s_2) - 1$ for all s_1, s_2 such that $s_1 \models \phi$ and $s_2 \not\models \phi$
  else
    $w(s_1, s_2) \leftarrow w(s_1, s_2) + 1$ for all s_1, s_2 such that $s_1 \models \phi$ and $s_2 \not\models \phi$
  end if
  $R = R - (\phi, i)$.
end while
Return $\langle S, w \rangle$.

in Present*, and that the report history R consists of the following reports:*

$$\langle A, 1 \rangle, \langle A, 1 \rangle, \langle B, 0 \rangle, \langle A \wedge B, 1 \rangle$$

Since our report history has size 4, the initital trust graph would look like Figure 1, except that all edge weights would be 4. After the first report, the edge weights would be increased on the following edges:

$$(\{A, B\}, \emptyset), (\{A, B\}, \{B\}), (\{A\}, \emptyset), (\{A\}, \{B\}).$$

The same thing would happen after the second report. On the third report, we would subtract one from the following edges:

$$(\{A, B\}, \emptyset), (\{A, B\}, \{A\}), (\{B\}, \emptyset), (\{A\}, \{B\}).$$

Finally, the fourth report would add one to the following edges:

$$(\{A, B\}, \emptyset), (\{A, B\}, \{A\}), (\{A, B\}, \{B\}).$$

The final trust graph is given in Figure 2. Based on this graph, Present *is least trusted to distinguish the states $\{B\}$ and $\emptyset$. This is because the positive reports were all related to the truth of A, and the only false report was a report about the trust of B. Hence, the graph is intuitively plausible.*

3.3. Basic Results

We have defined an approach to building trust graphs from reports. We remark that the edge weights will not be used directly when it comes to belief revision. For belief revision, what we need is a single *trust ranking* that is derived from the trust graph. However, constructing the graph allows us to define the ranking function as sort of a consequence of the reports. In this section, we show the construction satisfies some desirable properties.

First, we define the trust ranking associated with a trust graph.

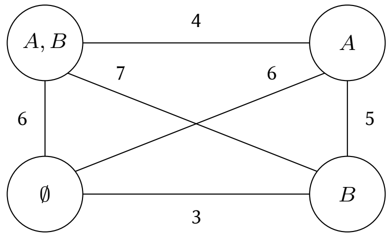

Figure 2: Graph Construction

Definition 6. *For any trust graph $T = \langle S, w \rangle$, let d_T denote the minimax distance between states.*

The distance d_T captures an overall trust ranking that can be used to inform belief revision. Informally, even if an agent is not trusted to distinguish two states directly, they may be trusted to distinguish them based on a path in the graph. The important feature of such a path is the minimax weight. The following is a basic result about the notion of distance defined by a trust graph.

Proposition 2. *For any trust graph $T = \langle S, w \rangle$, the function d_T is a pseudo-ultrametric on S.*

Recall from Section 2 that a pseudo-ultrametric over states can be used to define a ranking that is suitable for reasoning about trust. We remark that, in fact, every ultrametric over a finite set is actually equivalent up to isomorphism to an ultrametric defined by the minimax distance over some weighted graph. This means that every trust ranking can be defined by a trust graph.

The next result shows that there is nothing particularly special about the trust graphs constructed by our algorithm.

Proposition 3. *Every weighted graph over S is the trust graph obtained from some report history R.*

This can be proven by a simple construction where each report only modifies a single edge weight.

In the next results, we adopt some simplifiying notation. If R is a report history and r is a report, we let $R \cdot r$ denote the multiset obtained by adding r to R. Also, if R is a report history, we let $T(R)$ denote the trust graph obtained from R and we let d_R denote the distance d defined by $T(R)$.

As stated, Algorithm 1 can only construct a trust graph starting from scratch. However, the following proposition states that we can iteratively modify a trust graph as we get new reports.

Proposition 4. *Let R be a report history and let r be a report. Then $T(R \cdot r)$ is identical to the trust graph obtained by modifying $T(R)$ as follows:*

- *Increment weights between states that disagree on ϕ, if r is a positive report.*
- *Decrement weights between states that disagree on ϕ, if r is a negative report.*
- *Defining a new minimax distance d in accordance with the new edge weights.*

Hence, rather than viewing trust graphs as something created with no a priori knowledge, we can think of trust graphs as a simple model of trust together with an operation that tweeks the weights to respond to a new report.

One desirable feature of our construction is that a report of $(\phi, 0)$ should make the reporting agent less trustworthy with regards to reports about the trust of ϕ. The next proposition shows that this is indeed the case.

Proposition 5. *Let R be a report history, let s_1 and s_2 be states such that $s_1 \models \phi$ and $s_2 \not\models \phi$. Then*

$$d_R(s_1, s_2) \geq d_{R\cdot(\phi,0)}(s_1, s_2).$$

We have a similar result for positive reports.

Proposition 6. *Let R be a report history, let s_1 and s_2 be states such that $s_1 \models \phi$ and $s_2 \not\models \phi$. Then*

$$d_R(s_1, s_2) \leq d_{R\cdot(\phi,1)}(s_1, s_2).$$

Taken together, these results indicate that negative (resp. positive) reports of ϕ make the reporting agent less (resp. more) trusted with respect to ϕ. We remark that the inequalities in the previous theorems would be strict if we were considering actual edge weights; but they are not strict for d_R, since there may be multiple paths between states.

We have seen that trust graphs define a distance over states that represents a general notion of trust that is implicit in the graph. Significantly, trust graphs can be constructed in a straightforward way by looking at past reports; the implicitly defined trust ranking is based on the accuracy of these reports. In the next section, we consider how the notion of trust defined by a trust graph can be used to construct different approaches to revision.

4. Using Trust Graphs

4.1. Example Revisited

Consider again our example involving reports about the lights in a building. We previously pointed out that *Absent* might not actually trust the reports from *Present*, and we gave an approach to construct a trust graph.

Informally, when talking about trust, we might make assertions of the following form:

1. *Present* is not trusted to know which room they are in.
2. *Present* is not trusted to check two rooms at once.

These kind of assertions give us a hint about how belief revision might occur. For example, in the first case, *Absent* would interpret a report to mean that *exactly one* of the rooms is lit.

Note, however, that a trust graph does not simply give a binary notion of trust; it defines a distance function that indicates strength of trust in various distinctions. Similarly, the beliefs of an agent might be held with different levels of strength. So, even if we have a trust graph, there are still problems with incorporating reports related to comparing *strength of belief* with *strength of trust*.

In our example, if *Absent* just left the building, they might believe *very strongly* that the light in room A must be off. They might believe this so strongly that they disregard *Present*'s report entirely. But disregarding reports is not the only option. It might be the case that the exact strength of *Absent*'s beliefs needs to be considered. Suppose *Absent* believes the light in room A is off with a *medium* degree of strength. In that case, a report from a weakly trusted agent will not change their beliefs, whereas a report from a strongly trusted agent would be more convincing. Moreover, *Absent* also needs to have a strength ranking over possible alternatives. Hence, this is not simply a binary comparison between strength of degree and strength of trust. In order to model interaction between belief and trust, we need a precise formal account that permits a comparison of the two. We also need to account for the way that *Present* develops a reputation, either for laziness or inaccuracy.

4.2. Naive Revision with Trust Graphs

In the remainder of this paper, we assume that the beliefs of an agent are represented by an OCF. We show how a trust graph allows us to capture an approach to belief revision that takes trust into account. In fact, the approach in this section depends only on a pseudo-ultrametric d_T defined by a trust graph.

For any pseudo-ultrametric d, we define a family of revision operators $*_n$.

Definition 7. *Let κ be an OCF and let d be a pseudo-ultrametric over S. For each n, the operator $*_n$ is defined such that $Bel(\kappa) *_n \phi$ is equal to:*

$$\min_{\kappa}\{s \mid \textit{there exists } t \textit{ such that } d(t,s) \leq n \textit{ and } t \models \phi\}$$

From the theory of metric spaces, we have the following.

Proposition 7. *For any pseudo-ultrametric d over a set X, if $n \in \mathbf{N}$ then the collection of sets $Y_x = \{y \mid d(x,y) \leq n\}$ is a partition of X.*

The next result relates these revision operators to trust-sensitive revision operators. A parallel result is proved in [9], although the result here is stated in terms of OCFs rather than AGM revision.

Proposition 8. *Let κ be an OCF and let T be a trust graph. For any formula ϕ and any n:*

$$Bel(\kappa) *_n \phi = Bel(\kappa) *^{\Pi} \phi$$

where Π is the partition defined by (d_T, n) and $*^{\Pi}$ is the trust-senstive revision operator associated with Π.

Hence κ and d_T define a set of trust-sensitive revision operators. The parameter n specifies how close two states must be to be considered indistinguishable in the partition.

We refer to the operators $*_n$ as *naive* trust-sensitive revision operators in this paper. These operators are naive in the sense that they do not allow us to take into account the relative magnitudes of the values in κ and the distances given by d_T. In other words, the scales of κ and d_T are not compared; it doesn't matter if the initial strength of belief is high or low. This makes sense in applications where the magnitudes in κ and d_T are seen as independent.

Example 3. *We refer back to our motivating example. Suppose that the initial beliefs of* Absent *are given by κ such that:*

$$\begin{aligned} \kappa(\{A\}) &= 0, \\ \kappa(\{B\}) &= 1, \\ \kappa(\{A,B\}) &= 1, \\ \kappa(\emptyset) &= 2 \end{aligned}$$

Hence the initial belief set for Absent *is $\{A\}$. Now suppose that* Present *passes a message that asserts $\neg A \wedge B$; in other words, the light is off in A while it is on in B. If this information was given to* Absent *as infallible sensory data, then the result could be determined easily with regular AGM revision. But this is not sensory data; this is a report, and trust can play a role in how it is incorporated.*

To make this concrete, suppose that Absent *thinks that* Present *is generally lazy and unaware of the room that they are in. It is unlikely therefore, that* Present *would run quickly from one room to another to verify the status of the light in both. So perhaps the trust graph T constructed from past reports defines the distance function d_T from $\{B\}$ as follows:*

$$\begin{aligned} d_T(\{B\},\{A\}) &= 1 \\ d_T(\{B\},\{B\}) &= 0 \\ d_T(\{B\},\{A,B\}) &= 10 \\ d_T(\{B\},\emptyset) &= 5 \end{aligned}$$

This distance function does indeed encode the fact that Present *is not strongly trusted to distinguish $\{A\}$ and $\{B\}$; this is because they do not always know where they are.*

We have supposed that Present *reports $\neg A \wedge B$. So, what should* Absent *believe? It depends on the threshold n. If we set $n = 3$, then by Proposition 6, $*_3$ is the trust-sensitive revision operator defined by the partition with cells $\{\{A\},\{B\}\}$ and $\{\{A,B\},\emptyset\}$. Since $\{A\}$ and $\{B\}$ are in the same cell, it follows that revision by B is equivalent to revision by $A \vee B$. Hence:*

$$Bel(\kappa) *_3 B = \{\{A\}\}.$$

This is a belief state containing just one state; so Absent *believes that the most plausible state is the unique state where only the light in room A is on. Hence, if* Present *reports that the light in room B is on, it will not change the beliefs of A at all.*

For naive operators, it does not matter how strongly *Absent* believes the light in room A is on. It only matters whether or not the reporting agent can distinguish between particular states.

4.3. General Revision with Trust Graphs

In the previous section, we considered the case where strength of belief and strength of trust are incomparable; the magnitudes of the values are not on the same scale. In this case, we can not meaningfully combine the numeric values assigned by κ with the numeric distances given by a trust graph; we essentially have two orderings that have to be merged in some way. This is the general setting of AGM revision, and trust-sensitive revision.

However, there is an alternative way to define revision that actually takes the numeric ranks into account. First, we define a new OCF, given some initial beliefs and a trust distance function.

Definition 8. *Let κ be an OCF and let d be a pseudo-ultrametric. For any $s \in S$:*

$$\kappa_d^{\phi}(s) = \kappa(s) \cdot \min\{d(s,t) \mid t \models \phi\}.$$

The OCF $\kappa_d^{\phi}(s)$ combines the a priori belief in the *likelihood* of s along with a measure indicating how easily s can be distinguished from a model of ϕ. Essentially, this definition uses d to construct a ranking function over states centered on $|\phi|$. This ranking is aggregated with κ, by adding the two ranking functions together.

Given this definition, we can define a new revision operator.

Definition 9. *Let κ be an OCF and let d be a pseudo-ultrametric. For any formula ϕ, define $\circ_d$ such that*

$$Bel(\kappa) \circ_d \phi = \{s \mid \kappa_d^{\phi}(s) \text{ is minimal}\}.$$

This new definition lets the initial strength of belief be traded off with perceived expertise. We return to our example.

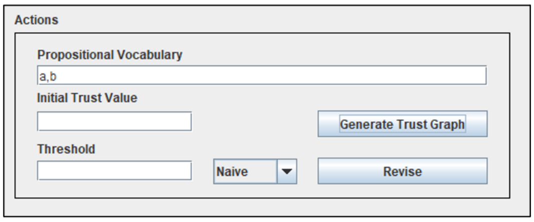

Figure 3: Initializing a Trust Graph

Example 4. *Consider the light-reporting example again, with the initial belief state κ and the distance function d_T specified in Example 3. Now suppose again that* Present *reports $\phi = \neg A \wedge B$, i.e. that only the light in room B is on. We calculate $\kappa_d^{\phi}(s)$ for all states s in the following table.*

s	$\kappa(s)$	$d(\{B\}, s)$	$\kappa_d^{\phi}(s)$
$\{A\}$	0	1	1
$\{B\}$	1	0	1
$\{A, B\}$	1	10	11
$\emptyset$	2	5	7

Since the first two rows both have minimal values, it follows that

$$Bel(\kappa) \circ_d * \neg A \wedge B = \{\{A\}, \{B\}\}.$$

Following revision, Absent *believes exactly one light is on.*

This example demonstrates how the strength of belief and the strength of trust can interact. The given result occurs because the strength of belief in $\{A\}$ is identical to the strength of trust in the report of $\{B\}$. Increasing or decreasing either measure of strength will cause the result to be different. Note also that this approach gives a full OCF as a result, so we have a ranking of alternative states as well.

5. Implementation

5.1. Functionality

We describe *T-BEL* , a Java application for modeling the dynamics trust and belief. The core functionality of *T-BEL* is as follows. It allows a user to create a trust graph that captures the distinctions an information source is trusted to make. It allows a user to enter a series of reports, which might be correct or incorrect. These reports trigger an update to the trust graph. Finally, the user can calculate the result of belief revision, in a manner that accounts for the influence of trust.

Note that the steps listed above need not be done sequentially. The interface for the software provides several panels for different actions: initializing a trust graph, manipulating the trust graph, visualizing the trust graph, and performing revision. The only constraint is that the vocabularly needs to be provided to initialize the trust graph. After the initial trust graph is constructed, a user can jump between different panels. For example, one could add new information about past reports at any time, even after revision has been performed.

In the following sections, we describe the basic usage of the software.

5.2. Constructing a Trust Graph

In order to perform belief revision using *T-BEL* , we first need to initialize a trust graph. This is done through the panel in Figure 3. The user simply enters a propositional vocabulary as a comma delimited sequence of strings. Optionally, one can specify an initial trust value; this is the weight that will be assigned to all edges in the trust graph. If it is not specified, it will default to 1.

The panel in Figure 4 is used for visualizing and manipulating the trust graph. After the trust graph has been generated, it is displayed on the left side as a matrix that gives the weight between every pair of states. The values in this matrix can be edited manually, but this is not the preferred way to change the values. The main goal of *T-BEL* is to allow trust to be built incrementally by adding reports. This is done through the *report entry* section in Figure 4. Reports are entered as formulas in a simple variant of propositional logic, using the keyboard-friendly symbols & (conjunction), | (disjunction) and − (negation). The reports are tagged with 1 (positive) and 0 (negative). By default, when the *Add Reports* button is pressed, the matrix on the left updates the values in accordance with the following update rules:

Update Rule 1. *For each pair of states s_1, s_2 such that $s_1 \models \phi$ and $s_2 \not\models \phi$ decrease the value $w(s_1, s_2)$ to $w(s_1, s_2) - 1$.*

Update Rule 2. *For each pair of states s_1, s_2 such that $s_1 \models \phi$ and $s_2 \not\models \phi$, increase the value $w(s_1, s_2)$ to $w(s_1, s_2) + 1$.*

These update rules correspond to the construction of a trust graph in Algorithm 1. However, we remark that *T-BEL* is not restricted to these updates. If the user would like to specify different update rules, this can be done by providing a text file specifying new update rules.

There is one remaining feature to mention in this panel: the *Distance Checker.* We will see in the next section that we actually do not use the values in the trust matrix directly; we use the minimax distance generated from these values. As such, we provide the user with a simple mechanism for checking minimax distance for testing and experimentation.

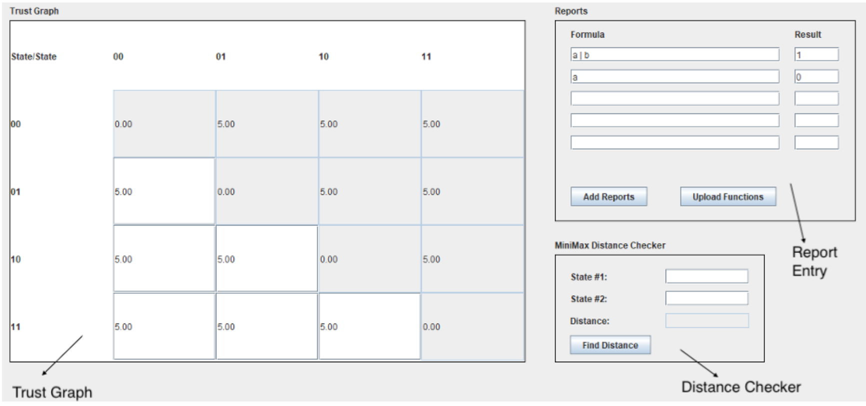

Figure 4: The Trust Panel

5.3. Specifying an Epistemic State

As noted previously, epistemic states are represented in *T-BEL* using ranking functions. The software provides two different ways to specify an epistemic state.

The first way to specify an epistemic state is by explicitly specifying a total pre-order over all states. This is done by creating an external text file that lists a "level" for all states starting from 0. For example, if we had two variables A and B, then one example input file could be specified as follows:

```
2
0 : 00
1 : 10
```

The first line indicates that there are 2 variables. The second line says that the state where A and B are both false is the most plausible, so it is the only state in level 0. The next line specifies the states in level 1. Any states not listed are put in level 2. A ranking over states specified in this manner gives us enough information to perform belief revision.

Manually specifying a complete ranking in this manner can be problematic, because it is time consuming and it is easy to make mistakes. As such, we also give the user the ability to experiment with revision simply by entering a belief state as a set of formulas through an input box in the main interface. For example, we could enter the beliefs by giving this list of formulas:

```
A&B
A | -B
```

To generate a ranking function from such a list, *T-BEL* finds all satisfying assignments. In the example given, the only satisfying assignment occurs when A and B are both true. By default, *T-BEL* then uses the Hamming distance from the set of satisfying assignments to create a full ranking. In other words, the default approach defines a ranking that corresponds to Dalal's revision operator [10]. However, *T-BEL* also provides a flexible mechanism for reading alternative rankings from a file input.

5.4. Calculating the Result of Revision

T-BEL implements both naive revision and general revision; the user chooses the mechanism to be used in the menu in Figure 3.

If *Naive Revision* is selected, then the user needs to enter a threshold value. Following Proposition 8, this threshold value defines a trust-sensitive revision operator. This operator is used to calculate the result of belief revision when the Naive option is selected. The result of revision is displayed as a formula, capturing the minimal states in the new ranking. We note that the software can be used to perform more than one revision when the Hamming distance has been specified for the ranking. However, for file-based rankings, iterated revision is not possible.

We can also specify that we want to use general revision in the dropdown menu in Figure 3. In this case, if κ is the original ranking function, d is the minimax distance and ϕ is the formula for revision, then we can define a new function:

$$\kappa_d^{\phi}(s) = \kappa(s) + \min\{d(s,t) \mid t \models \phi\}.$$

By normalizing this function, we define a new ranking function that represents the beliefs following revision. The result of belief revision is displayed as a formula. However, general revision can be iterated because the full ranking is maintained after each revision.

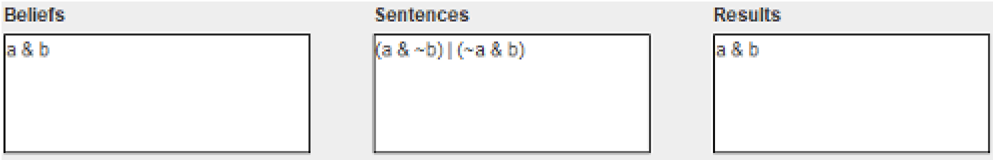

Figure 5: Revision Output

5.5. Step by Step Example

Assume we want to work with the vocabularly $\{a, b\}$, as well as past reports of $(a \vee b, 1)$ and $(a, 1)$. Assume further that we would like to start with the belief state $(a \wedge b)$ and then revise by $(a \wedge \neg b) \vee (\neg a \wedge b)$. Using *T-BEL* , then can solve this problem through the following steps:

1. Enter the vocabulary a, b and a default value of 5.
2. Enter reports $(a|b, 1)$ and $(a, 1)$ then click *Add Reports.*
3. Select Naive revision with threshold 3.
4. Enter the belief state $a\&b$ and formula $(a\& -b)|(-a\&b)$.
5. Click *Revise.*

The default value in step 1 should be set so that it is at least as high as the number of reports. However, beyond that constraint, it will not impact the results. After step 2, the values in the matrix representing the trust graph will be as follows:

	00	01	10	11
00	0	6	7	7
01	6	0	6	6
10	7	6	0	5
11	7	6	5	0

The revision panel following the example is in Figure 5, showing the input and the fact that the beliefs are unchanged after revision. It can easily be verified that this is correct.

5.6. Performance

The question of run time is a challenging one to address for any implemented belief revision system, due to the well known compexity of revision [11]. The problem is even worse when we add trust graphs, which become very large as the vocabulary size increases.

The present implementation has made many implementation choices in order to optimize performance. For example, we represent a trust map internally as a hashmap of hashmaps; the lookup time is very fast. Another place where we focus on efficiency is in the translation from formulas to belief states, where we use a DPLL solver to find satisfying assignments. However, the run time for *T-BEL* still becomes slow as the vocabulary size increases. It is a useful prototype for reasoning about small examples, and demonstrating the utility of trust graphs. In future work, we will look to improve run time by integrating a competition level ALLSAT solver for the hard calculations [12].

6. Discussion

6.1. Related Work

This work fits in the general tradition of formalisms that address notions of trust and credibility for belief revision. There are alternative approaches, based on *non-prioritized* and *credibility-limited* revision as well [13, 14, 15]. The notion of trust has been explored in the setting of Dynamic Epistemic Logic (DEL), by adding an explicit measure of trust to formulas [16]. Finally, since we are primarily concerned with with trust based on expertise, the formalism presented here is also related to recent work on truth discovery [17].

But fundamentally, this work is really about building trust in a source based on the knowledge demonstrated in past reports; our goal is to develop a formal model of knowledge-based trust. To the best of our knowledge, this problem has not been explored previously in the context of formal belief change operators. However, it has been explored in some practical settings, such as the formulation of search engine results [18].

The software introduced here can be seen as an extension of the GenB system [19]. GenB is a general solver for revision with a limited capacity to capture trust; *T-BEL* is significantly more sophisticated when it comes to representing and reasoning about the dynamics of trust and belief.

6.2. Conclusion

In this paper, we have addressed the problem of building trust from past reports. We demonstrated that, in the context of OCFs, trust can be interpreted in two ways. First, if the scale used for the the strength of belief is deemed to be independent of the distance metric, then we can use a trust ranking to define a family of *naive* revision operators for trust-sensitive revision. On the other hand, if strength of trust and strength of belief are considered to be comparable on the same scale, then we have shown how the two can be aggregated to define a new approach to trust-influenced belief revision.

We have also described a tool for solving belief change problems influenced by trust. The focus is on building trust from reports, and then performing belief revision. Our software provides a simple interface that can be used to build a trust graph iteratively, and then this graph is used to adjust the behaviour of a formal belief change operator to account for trust. We suggest that this tool is an important step towards demonstrating the utility of belief change operators for solving practical problems with partially trusted information sources.

There are many directions for future research. Beyond expanding the formal theory, we are primarily interested in practical applications of this work. In future work, we intend to improve run time performance, apply the tool to concrete problems in the evaluation of web resources, and connect our approach to related work on learning with respect to trust.

References

[1] A. Hunter, Building trust for belief revision, in: Proceedings of the Pacific Rim Conference on Artificial Intelligence (PRICAI), 2021, pp. 543–555.

[2] C. E. Alchourrón, P. Gärdenfors, D. Makinson, On the logic of theory change: Partial meet functions for contraction and revision, Journal of Symbolic Logic 50 (1985) 510–530.

[3] W. Spohn, Ordinal conditional functions. A dynamic theory of epistemic states, in: W. Harper, B. Skyrms (Eds.), Causation in Decision, Belief Change, and Statistics, vol. II, Kluwer Academic Publishers, 1988, pp. 105–134.

[4] M. Carbone, M. Nielsen, V. Sassone, A formal model for trust in dynamic networks, in: International Conference on Software Engineering and Formal Methods, 2003, pp. 54–61.

[5] K. Krukow, M. Nielsen, Trust structures, International Journal of Information Security 6 (2007) 153–181.

[6] T. D. Huynh, N. R. Jennings, N. R. Shadbolt, An integrated trust and reputation model for open multi-agent systems, Autonomous Agents and Multi-Agent Systems 13 (2006) 119–154.

[7] A. Salehi-Abari, T. White, Towards con-resistant trust models for distributed agent systems, in: Proceedings of the 21st International Joint Conference on Artificial Intelligence (IJCAI), 2009, pp. 272–277.

[8] R. Booth, A. Hunter, Trust as a precursor to belief revision, J. Artif. Intell. Res. 61 (2018) 699–722.

[9] A. Hunter, R. Booth, Trust-sensitive belief revision, in: Q. Yang, M. J. Wooldridge (Eds.), Proceedings of the Twenty-Fourth International Joint Conference on Artificial Intelligence, IJCAI 2015, Buenos Aires, Argentina, July 25-31, 2015, AAAI Press, 2015, pp. 3062–3068.

[10] M. Dalal, Investigations into a theory of knowledge base revision, in: Proceedings of the National Conference on Artificial Intelligence (AAAI), 1988, pp. 475–479.

[11] T. Eiter, G. Gottlob, On the complexity of propositional knowledge base revision, updates and counterfactuals, Artificial Intelligence 57 (1992) 227–270.

[12] T. Toda, T. Soh, Implementing efficient all solutions sat solvers, ACM Journal of Experimental Algorithmics 21 (2016) 1–44.

[13] S. O. Hansson, E. L. Fermé, J. Cantwell, M. A. Falappa, Credibility limited revision, J. Symb. Log. 66 (2001) 1581–1596.

[14] R. Booth, E. Fermé, S. Konieczny, R. P. Pérez, Credibility-limited revision operators in propositional logic, in: Principles of Knowledge Representation and Reasoning: Proceedings of the Thirteenth International Conference, KR 2012, Rome, Italy, June 10-14, 2012, 2012.

[15] G. Bonanno, Credible information, allowable information and belief revision - extended abstract, in: L. S. Moss (Ed.), Proceedings Seventeenth Conference on Theoretical Aspects of Rationality and Knowledge, TARK 2019, Toulouse, France, 17-19 July 2019, volume 297 of *EPTCS*, 2019, pp. 82–90.

[16] F. Liu, E. Lorini, Reasoning about belief, evidence and trust in a multi-agent setting, in: PRIMA 2017: Principles and Practice of Multi-Agent Systems - 20th International Conference, Nice, France, October 30 - November 3, 2017, Proceedings, volume 10621 of *Lecture Notes in Computer Science*, Springer, 2017, pp. 71–89.

[17] J. Singleton, R. Booth, An axiomatic approach to truth discovery, in: Proceedings of the International Conference on Autonomous Agents and Multiagent Systems (AAMAS), 2020, pp. 2011–2013.

[18] X. Dong, E. Gabrilovich, K. Murphy, V. Dang, W. Horn, C. Lugaresi, S. Sun, W. Zhang, Knowledge-based trust: Estimating the trustworthiness of web sources, Proceedings of the VLDB Endowment 8 (2015).

[19] A. Hunter, E. Tsang, GenB: A general solver for AGM revision, in: Proceedings of the European Conference on Logics in Artificial Intelligence (JELIA), 2016, pp. 564–569.