Inductively Generated Trust Alignments based on Shared Interactions [*]

(Extended Abstract)

Andrew Koster
IIIA-CSIC, Spain
andrew@iiia.csic.es

Jordi Sabater-Mir
IIIA-CSIC, Spain
jsabater@iiia.csic.es

Marco Schorlemmer
IIIA-CSIC, Spain
marco@iiia.csic.es

ABSTRACT

In open multi-agent systems trust models are an important tool for agents to achieve effective interactions. However, the agents do not necessarily use similar trust models, leading to semantic differences between trust evaluations in the different agents. We show how to form a trust alignment by considering the interactions agents share. We describe a method, using inductive learning algorithms, to accomplish this alignment.

Categories and Subject Descriptors

Computing Methodologies [**Artificial Intelligence**]: Distributed Artificial Intelligence—*Multiagent Systems*

General Terms

Algorithms

Keywords

Trust and reputation, Learning, Ontological reasoning

1. INTRODUCTION

Intelligent agents have the ability to act autonomously and engage in social activities. Especially in open multi-agent systems these social activities may expose them to risks. A way for reducing this risk is finding the right partner in their interactions. One proposed solution to this problem is based on the concepts of trust and reputation to create a network of social control for the agents. There are already quite a large number of computational models for trust and reputation in use, each with a slightly different interpretation of what trust means. One of the major benefits proposed by trust models, is that the trust evaluation is communicable: agents can warn each other for fraudulent agents or help each other in their selection of a good partner. However, if the different agents use diverse models of trust, this communication becomes problematic. What does it mean to one agent when another agent communicates a trust evaluation?

Human trust is a social evaluation and as such cannot be seen independent from the social interactions which support that trust. The same holds for computational agents: trust is based on interactions between agents. These interactions are observed by different agents and the observations lead to trust evaluations of the various agents involved. If the agents use different trust models, the trust evaluation of these agents, based on the same observations, can also be different: trust *means* something different to the agents.

The interactions trust is based on can have many different forms, such as playing squash with someone, buying a bicycle on eBay or communicating trust evaluations. These interactions are observed by any number of agents, however the amount as well as type of information observed by these agents may be different. Additionally, agents may associate different subjective observations with the interaction. For instance, the seller in an eBay auction may not be satisfied with the transaction because he had to sell at a loss. This type of observation is private and often just as subjective as the trust evaluation itself. This difference in observations complicates the matter of aligning trust models, however we postulate that there is always some amount of shared information. At the very least, there is shared information that an interaction took place. Our approach uses these shared interactions as building blocks for a trust alignment.

2. THE ALGORITHM

We describe a mathematical framework for Trust Alignment in [2]. The intuition is that both agents can relate each others' subjective trust evaluations, communicated in the language $\mathcal{L}_{Trust}$, to the objective descriptions of interactions in $\mathcal{L}_{Domain}$. By doing so they are able to find the underlying meaning of trust evaluations. We translate the theory into a computational method for alignment. First the agents have to communicate their trust evaluations to each other in the form of messages, which describe their trust evaluation of target agents, based on some specific set of interactions. These interactions must be part of the set of interactions which both agents can observe. This allows the agents to align their subjective trust evaluations, by communicating objective information about the interactions these evaluations are based on.

For each message, the receiving agent computes its own trust evaluation, leading to a set of Specific Rules for Alignment (SRAs), each of the form $\alpha_i[T_j] \leftarrow \beta_i[T_j], \psi_i$, which would be the ith SRA about the target agent T_j. The heads of the rules α are the own trust evaluations, while β in the bodies are the other agent's. ψ describes the set of interactions which support both evaluations. The agent then has to learn the model underlying these SRAs. The output will be a set of General Rules for Alignment (GRAs), which is the

[*] This work is supported by the Generalitat de Catalunya grant *2009-SGR-1434* and the Agreement Technologies Project *CONSOLIDER CSD2007-0022, INGENIO 2010*

Cite as: Inductively Generated Trust Alignments based on Shared Interactions (Extended Abstract), A. Koster, J. Sabater-Mir and M. Schorlemmer, *Proc. of 9th Int. Conf. on Autonomous Agents and Multiagent Systems (AAMAS 2010)*, van der Hoek, Kaminka, Lespérance, Luck and Sen (eds.), May, 10–14, 2010, Toronto, Canada, pp. . 1571-1572

Algorithm 1: Generalize SRAs

```
   Input: $\mathcal{R}$, the set of SRAs to be generalized
   Input: $D(x, y)$, a distance measure on $\mathcal{L}_{Trust}$
   Input: S, a set of increasing distances for clustering
1  GRAs := $\emptyset$
2  Clusters := $\{\{r\}|r \in \mathcal{R}\}$
3  Covered := $\emptyset$
4  foreach Stop_criteria s in S do
5      Clusters := agglomerative_clustering(Clusters, s, D)
6      if |Clusters| = 1 then
7          break
8      foreach $C \in$ Clusters do
9          H := generalize_head(C, $\mathcal{R}\backslash$C)
10         if $H \neq null$ then
11             G := generalize_body(C, $\mathcal{R}\backslash$C)
12             if $G \neq null$ then
13                 GRAs := GRAs $\cup$ $\{\langle$H $\leftarrow$ G, s$\rangle\}$
14                 Covered := Covered $\cup$ C
15     if Covered = $\mathcal{R}$ then
16         break
17 Output: GRAs
```

generalization of the SRAs we give as input. The procedure used can be seen in Algorithm 1. We use three important procedures, which we will explain in more detail: the clustering algorithm in line 5 and the two generalization algorithms we use in lines 9 and 11.

2.1 Clustering

We consider those SRAs where the *receiving* agent's trust evaluations are "near each other", because we want to learn generalizations that will predict that agent's trust evaluations, based on the gossip sent. That means we cluster based on the heads of the SRAs and we have a requirement for $\mathcal{L}_{Trust}$: there must be a distance defined on it, with which we can incrementally cluster the SRAs. We have the following criteria for our clustering:

- We want to work our way from small precise clusters to large clusters, which cover a broad spectrum of trust evaluations.
- We want to be able to stop the algorithm when we have found GRAs covering all SRAs.

Bottom-up incremental clustering algorithms fit these criteria best. We stop the clustering process for each stop criterion s. These are defined by the programmer and form a list of maximum distances for the clustering algorithm. For each criterion s, the algorithm continues merging clusters until all the clusters are at a distance greater than s. It then moves to the next step.

2.2 Learning rules

For each stop criterion s we will have a set of clusters of SRAs and for each of these we shall attempt to generalize a set of GRAs covering it. Our first task is to learn a generalization of the heads of the SRAs. All the α_i within a cluster are within distance s of each other and we want to find some defining quality of these α_i, which we can use in our final ruleset. We want to learn the generalization α^* which θ-subsumes [1] all α_i. Afterwards, when we generalize the body, we are learning the conditions for which the receiving agent should have trust evaluation α^*.

2.2.1 Learning the head

Firstly we note that each α_i has some target agent T_j. We will immediately replace all these agents with a variable, because we want the resulting generalization to be independent of the agents evaluated. The centre of the cluster will be the least general generalization of the α_i under θ-subsumption. It is relatively easy to compute using an inductive learning algorithm with the "learn from example" setting [1]. We want to learn some phrase α^* in $\mathcal{L}_{Trust}$ such that if α^* holds then all α_i hold. Thus α^* should be a statement in $\mathcal{L}_{Trust}$ and we use this language to define the type of concept that should be learned. Inductive learning algorithms learn from a set of positive and negative examples of the concept to be learned. In our case the positive examples are the α_i in the cluster we want to generalize and all heads of SRAs that are outside the cluster are negative examples. The algorithm will now search for the most general generalization of the positive examples, which does not cover any of the negative examples. The result is α^*.

2.2.2 Learning the body

The main task is to learn the generalization of the bodies. We rewrite our SRAs with α^* in the head, such that we have a list of Clustered Alignment Rules: $\alpha^*[X] \leftarrow \beta_i[X], \psi_i$. Our task is now to find a set of rules $\Psi[X]$, such that if some $\beta^*[X], \psi^* \in \Psi$ holds, then there is a $\beta_i[X], \psi_i$, which holds and the agent can conclude $\alpha^*[X]$. This can also be learned using an inductive learning algorithm. We use the Clustered Alignment Rules in the cluster as positive examples of the concept and all SRAs outside the cluster count as negative examples. We note that we have more information available than when we learned the generalization of the head, namely we have a list of situations β_i, ψ_i in which α^* holds, rather than just examples of the concept α^*. This coincides with the "learning from interpretation" setting of ILP [1], which allows for better heuristics than "learning from example", resulting in a faster algorithm for similar problems.

2.3 Forming the alignment

If we can find a generalization for the body it means we have a GRA which covers all of the SRAs in the cluster. We stop the algorithm when all SRAs are covered, or when the remaining clusters are further apart than the largest stop criterion. When the algorithm ends we have a list of GRAs. This list can be used to translate trust evaluations from the other agent. Because each GRA is stored with the stop criteria which allowed it to be generated, we have an internal distance of the cluster it covers. We use this as the measure of accuracy of the alignment. We can use this, together with the actual aligned message, in the trust model.

In addition to being inaccurate an alignment may be incomplete. Firstly, the interactions which lead to a trust evaluation in the other agent may not lead to any trust evaluation at all in the own agent. This is not a failure of the algorithm, but an indication that alignment is simply not possible. Secondly, the generalization algorithms may fail for a subset of the SRAs at each cluster distance, resulting in the alignment algorithm ending when the largest stop criterion is exceeded. This means we need more interactions to discover the alignment.

3. CONCLUSION

The method we propose addresses the problem of trust alignment in a novel way. A prototype implementation, which is outside the scope of this extended abstract, shows its viability. The key concept of our method is to use clustering and ILP to form the alignment of subjective trust evaluations using objective descriptions of the interactions.

4. REFERENCES

[1] L. De Raedt. *Logical and Relational Learning*. Springer Verlag, 2008.

[2] A. Koster, J. Sabater-Mir, and M. Schorlemmer. A formalization of trust alignment. In *Proc. of CCIA'09*, volume 202 of *Frontiers in AI and Applications*, pages 169–178, Cardona, Spain, 2009. IOS Press.