Discovering Frequent Patterns to Bootstrap Trust

Murat Sensoy[1,2], Burcu Yilmaz[1,3], and Timothy J. Norman[1]

[1] Department of Computing Science, University of Aberdeen, UK
[2] Ozyegin University, Istanbul, Turkey
[3] Gebze Institute of Technology, Kocaeli, Turkey
{m.sensoy,burcu,t.j.norman}@abdn.ac.uk

Abstract. When a new agent enters to an open multiagent system, bootstrapping its trust becomes a challenge because of the lack of any direct or reputational evidence. To get around this problem, existing approaches assume the same a priori trust for all newcomers. However, assuming the same a priori trust for all agents may lead to other problems like *whitewashing*. In this paper, we leverage graph mining and knowledge representation to estimate a priori trust for agents. For this purpose, our approach first discovers significant patterns that may be used to characterise trustworthy and untrustworthy agents. Then, these patterns are used as features to train a regression model to estimate trustworthiness. Lastly, a priori trust for newcomers are estimated using the discovered features based on the trained model. Through extensive simulations, we have showed that the proposed approach significantly outperforms existing approaches.

1 Introduction

In open systems like the Web, there is no central authority that monitors interacting agents and guarantees that every agent in the system behave as expected. For instance, a seller in an e-market place such as *ebay* may list a product at cheaper price but may not deliver the same product or any thing at all. This brings the necessity of agents to evaluate others and select the most trustworthy interaction partners among alternatives. While the word trust may have different definitions in different domains, we define it here pragmatically as the degree of belief or subjective probability, with which a trustor believes a trustee will perform as expected when relied upon [12].

Existing approaches for trust generally depend on an interaction history to compute trust of an agent x to another agent y. The interaction history contains the interactions between x and y (i.e., direct evidence) and their outcomes (e.g., *success* and *failure*). If the number of the direct interactions is not enough to compute trust with confidence, reputational evidence is used to compute the trustworthiness of y. The past interactions between other agents and y serve as reputational evidence. Based on direct and reputational evidence, statistical trust approaches like Beta Reputation System (BRS) [11] and $TRAVOS$ [17] is used to compute the trustworthiness of the agent y as the subjective probability that a future interaction with y would have desirable outcome. As the number of evidence increases, the accuracy of the estimated trustworthiness increases and the number of unsatisfactory interactions are minimised.

Although the existing approaches can accurately model the trust, they require repetitive interactions between the agents to build an interaction history. This requirement leads to two related problems: *bootstrapping* and *whitewashing*. The bootstrapping

L. Cao et al.: ADMI 2012, LNAI 7607, pp. 93–104, 2013.

problem arises when a new agent joins to the system. In this case, trust cannot be computed based on direct or reputational evidence. To deal with bootstrapping problem, the existing approaches assume a priori trust value for the newcomers. For instance, in BRS and $TRAVOS$, the a priori trust is 0.5, which means positive and negative outcome is equally likely. If the a priori trusts is high, the agents with bad reputations whitewash their bad reputation and enters to the society as a newcomer. On the other hand, if the a priori trust is too low, the newcomers may not have any chance to interact with others and build a good reputation.

Malicious agents may adopt certain behavioural patterns to achieve their goals. These patters may determine their choices and lead to the emergence of certain motifs in their relationships with other entities. Even if an agent whitewashes its identity and change/forge some of its observable attributes, the same motifs may be observed as long as the agent does not change its behaviour. For example, a malicious seller may change its identity and advertise a completely new profile whenever its reputation decreases. Although the *name*, *location*, *email*, and *web site* in the new profile are different from previous ones, all these email addresses and web sites could be hosted by the same or similar service providers (e.g., free hosting services). If malicious sellers have a tendency to bear the same or similar pattern in their profiles, we may build a stereotype such as "sellers using free hosting services are less trustworthy". This stereotype is not based on a simple attribute of seller (e.g., a specific email address), but an example of a complex feature discovered about the malicious sellers.

This paper proposes to discover and exploit the complex features (i.e., patterns) of agents [3] to estimate a priori trust for them. Knowledge about each known agent is represented in detail as a graph based on an ontology. Then, these graphs are mined to discover two groups of patterns that frequently appear in trustworthy and untrustworthy agents respectively. Lastly, the discovered patterns are used as features to train a regression model to estimate the a priori trust for the unknown agents. Through extensive simulations, we have showed that the proposed approach significantly outperforms existing trust approaches.

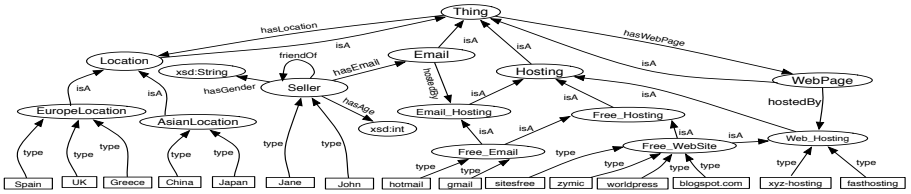

Fig. 1. A part of a simple ontology to describe sellers, their properties and relationships

2 Modelling Trust

Several approaches have been proposed to model trust in the literature [12]. A number of these approaches are based on *Subjective Logic* (SL), which is a belief calculus that allows agents to express opinions as degrees of belief, disbelief and uncertainty about propositions. Let ρ be a proposition such as "information source y is trustworthy in context c". Then, the binary opinion of agent x about ρ is equivalent to a Beta distribution. That is, the binomial opinion about the truth of a proposition ρ is represented as

the tuple (b, d, u, a), where b is the belief that ρ is true, d is the belief that ρ is false, u is the uncertainty, and a is base rate (a priori probability in the absence of evidence), as well as $b + d + u = 1.0$ and $b, d, u, a \in [0, 1]$. Opinions are formed on the basis of positive and negative evidences, possibly aggregated from different sources. Let r and s be the number of positive and negative past observations about y respectively, regarding ρ. Then, b, d, and u are computed based on Equation 1.

$$b = \frac{r}{r+s+2},\ d = \frac{s}{r+s+2},\ u = \frac{2}{r+s+2} \tag{1}$$

Then the opinion's probability expectation value is computed using Equation 2. Considering ρ, the computed expectation value can be used by x as the trustworthiness of y in the context c [12].

$$t^x_{y:c}(r, s, a) = b + a \times u = \frac{r + a \times 2}{r+s+2} \tag{2}$$

The base rate parameter a represents a priori degree of trust x has about y in context c, before any evidence has been received. The default value of a is mostly choose as 0.5 in literature [10], which means that before any positive or negative evidence has been received, both outcomes are equally likely. While x has more evidence to evaluate trustworthiness of y, the uncertainty u, so the effect of a, decreases.

For clarity, in this paper, we assume that the trust is computed by the same trustor agent x in the same context c. Therefore, we shortly use t_y instead of $t^x_{y:c}$ to represent trustworthiness of y in the context c for the agent x.

3 Knowledge Representation

To describe agents and their relationships semantically and flexibly, we propose to use an ontology [9]. To demonstrate toy examples in the paper, Figure 1 shows a part of a simple ontology, where arcs, ellipses, and rectangles represent relationships, concepts and their instances, respectively. Description of an agent is represented using $\langle subject, relation, object\rangle$ triples, such as $\langle john, hasLocation, Spain\rangle$. In these triples, subjects, objects, and properties are terms from the ontology. That is, subjects are instances of concepts (e.g., *john*); objects are literals (e.g., 60), concepts (e.g., *Location*), or their instances (e.g., *Spain*); and relations are datype properties (e.g. *hasAge*) or object properties (e.g., *hasLocation*) [9].

Each known agent is represented as an instance (e.g., instance of *Seller* concept) in the ontology and described using $\langle subject, relation, object\rangle$ triples. The description of the agent y can be semantically represented as a labelled directed graph G_y. The node representing y in G_y is called terminal node and connected to other nodes through relationships from ontology. Nodes in G_y refers to either literals or instances and edges of G_y correspond to relationships between those. A seller agent $john$ is represented as the graph shown in Figure 2. Given the ontology, this graph describes the seller $john$. However, it is not possible to completely interpret or reason about it without the ontology, since the terms (i.e., individuals and properties) used in this graph are defined within this ontology. That is why this graph is called *ontology-dependent* graph.

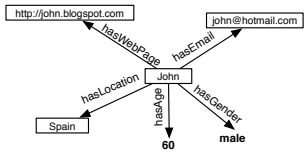

Fig. 2. Ontology-dependent description of the seller $john$

All most all of state of the art graph mining tools do not incorporate ontological knowledge [8]. Hence, they completely neglect the semantics of the labels in a graph. For instance, for these approaches, $Spain$ in the graph of Figure 2 is just a label of a node. However, the ontology of Figure 1 implies that $Spain$ is a location in Europe. Therefore, ontology-dependent graphs are not much informative for existing graph mining approaches [18]. One way of making them more informative as stand alone graphs is to embed relevant ontological knowledge into them. For this purpose, we use an ontological reasoner such as Pellet [16] and derive all direct and indirect statements (i.e., triples) about the individuals in the graph. Let us note that these individuals correspond to nodes in the original graph. Then, based on each triple $\langle subject, relation, object \rangle$, we create new nodes and relations if they are missing in the graph. As a result, the graph contains all direct and inferred information about the case at different levels of abstraction. Using individual's names as nodes' labels may hamper frequent pattern mining, since these individuals may not appear frequently in the dataset. To avoid this, we replaced these labels with "?" and added name (i.e., URI) of the referred individuals as a property to these nodes, i.e., using *name* datatype property. The resulting graph is called *ontology-independent* graph and referred to as g_y, because we do not need an ontology to infer properties of individuals on the graph; instead each node referring to an individual bares all direct and inferred attributes of the individual. Figure 3 shows ontology-independent version of the case graph shown in Figure 2. The terminal node is coloured *black* in the graph.

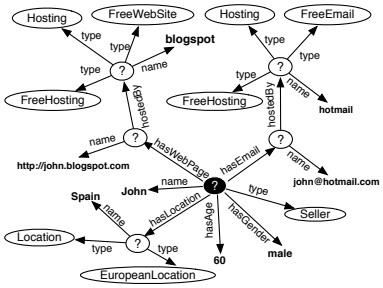

Fig. 3. Ontology-independent description of the seller $john$

Although we use a simple toy example to demonstrate the proposed graph-based representation here, this representation is very expressive and flexible enough to accommodate not only the attributes of the trustee but also all of its relationships with other entities (e.g., *friend of friend* relationships).

4 Graph Mining for Estimating a Priori Trust

In the previous section, we have described how the information about an agent can be represented as a graph, which can capture various aspects of the agent such as its attributes and even its relationships with other agents at different levels of abstraction. In this section, we propose to exploit graph mining techniques to discover frequent patterns that exist more frequently in either trustworthy or untrustworthy agents. For this purpose, we create graph datasets composed of graphs describing agents. Then, we use the discovered frequent patterns as features to estimate a priori trust for the agents.

4.1 Determination of Significant Patterns

For each known agent y, a graph g_y is generated to describe the agent. Then, g_y is labelled based on t_y; the labels are the categories based on the degree of trustworthiness. In this work, for the sake clarity and simplicity, we use only two categories: *trustworthy* and *untrustworthy*. Therefore, in the resulting graph data set, there are two classes of graphs: C_+ and C_-, which represent the graphs generated from *trustworthy* and *untrustworthy* agents respectively. An agent y is considered trustworthy if $t_y \geq \gamma$ and untrustworthy if $t_y \leq \delta$ where $\gamma > 0.5 > \delta$[1]. We want to discover two sets of patterns: P_+ and P_-, which are called patterns of *trust* and *distrust*, respectively. A pattern $p \in P_-$ is a subgraph that repeats frequently in the graphs of C_- and rarely in the graphs of C_+. Therefore, p distinguishes untrustworthy agents from trustworthy ones. Similarly, P_+ represents the significant patterns that repeat more frequently in the graphs of C_+. A formal definition of the patterns in P_- and P_+ is given in Definition 1 based on the frequencies explained in Definition 2.

Definition 1. Let $\nu \in \{+,-\}$ and $\bar{\nu} \in \{+,-\} \setminus \{\nu\}$. A frequent pattern $p \in P_\nu$ is a pattern such that $f(p, C_\nu) \geq \alpha_\nu > \sigma f(p, C_{\bar{\nu}})$, where $f(p, C_\nu)$ and $f(p, C_{\bar{\nu}})$ are frequencies of p in classes C_ν and $C_{\bar{\nu}}$, respectively; $0 < \alpha_\nu \leq 1$ is a threshold and $\sigma \geq 1$ is a coefficient. ∎

Definition 2. Given a pattern p, $f(p, C_\nu)$ is the frequency of the pattern in the graphs of class C_ν and computed by the formulae:

$$f(p, C_\nu) = \frac{N_{p:C_\nu}}{|C_\nu|}$$

where $N_{p:C_\nu}$ and $|C_\nu|$ are the number of graphs containing p in class C_ν and size of C_ν, respectively. ∎

Without support constraints (i.e., α_+ and α_-), we have to deal with huge number of patterns. To narrow down the search space, the trustor agent determines these constraints. An additional constraint can be the *minimum size* of the frequent patterns, where the size of the pattern is determined by its number of edges. In the literature, a number of *frequent subgraph mining* approaches have been proposed to extract frequent patterns in multi-class graph datasets using support and size constraints [5, 6, 18].

[1] To accommodate the blurred boundary between of trustworthiness and untrustworthiness, we set γ and δ to 0.75 and 0.4, respectively, in our evaluations.

4.2 Estimating a Priori Trust

As described in detail above, the trustor agent finds two sets of frequent patterns (P_+ and P_-) using frequent subgraph mining. In this section, we describe how these significant patterns can be used as features to estimate a priori trust for agents. Then, the estimated a priori trust is used as the base rate a in Equation 2 while computing the trustworthiness of the agents.

To estimate the a priori trust, the trustor can use various machine learning techniques [1]. In this work, we employ M5 regression tree algorithm [15] for this purpose. Given a training set and a set of features $\mathcal{F}$, this algorithm learns a mapping between feature values of an agent and its trustworthiness. In this paper, each feature $f_j \in \mathcal{F}$ corresponds to a specific pattern $p_j \in P_+ \cup P_-$ and its value for an agent i indicates if the graph g_i describing the agent entails p_j or not.

The training set is prepared using the descriptions of known agents and their trustworthiness. That is, for each known agent i, the training set contains a feature vector $\boldsymbol{v_i}$ and trust value t_i. Each field v_{ij} in the feature vector corresponds to the value of feature f_j for the agent i, which is determined by checking entailment of p_j by g_i. That is, $v_{ij} = 0$ if the patter p_j does not exist in the graph g_i describing i; otherwise $v_{ij} = 1$. In this way, we transform agents' graphs of various sizes into a vector dataset with fixed dimensions. After creating feature vectors for each known agent i, the set of $(\boldsymbol{v_i}, t_i)$ pairs are used as a training set to learn the regression model. The trained regression model is used as a function $R : \boldsymbol{v} \rightarrow [0, 1]$ that takes the feature vector $\boldsymbol{v}$ of an agent as input and returns an estimation of its trustworthiness. This estimation is not based on any evidence about the agent but only its features, hence what is estimated is actually the agent's a priori trust.

In order to compute trustworthiness of an unknown agent y, first the a feature vector $\boldsymbol{v_y}$ is created based on the discovered features. Then, t_y is computed using Equation 2 where $R(\boldsymbol{v_y})$ is used as the base rate a.

The trustor may have more interactions over time and learn trustworthiness of others better based on new evidence. Meanwhile, dynamics of the society may change and new behavioural patterns may be adopted by malicious agents. Therefore, the trustor may periodically use the described approach to discover new features significant for trustworthiness and retrain the regression model based on these new features.

5 Evaluation

In evaluating our approach, we employed a simulated agent society where a set of trustor agents interact with a set of trustee agents over a number of rounds. Each trustee is assigned a performance profile which determines how it will behave. Each profile specifies the mean and standard deviation parameters of a Gaussian distribution from which simulated interaction outcomes will be drawn. A trustor considers an interaction's outcome as a *success* if it is greater than a threshold λ; otherwise it is considered a *failure*. This threshold could vary for each trustor, so that different trustors may perceive the same outcome differently. However, for simplicity, we assume that all trustors use the same threshold value. Table 1 uses the values of all parameters used in the simulations.

Table 1. Experimental Parameters

Parameter	Value	Description
γ	0.75	Threshold for trustworthiness
δ	0.4	Threshold for untrustworthiness
$\alpha_- = \alpha_+$	0.1	Support constraints
σ	2	Frequency coefficient
$Life$	500	Simulation life
N_a	100	Number of trustee agents
N_t	20	Number of trustor agents
N_i	100	Total number of concept instances
N_{avail}	10	Number of available trustees
N_{rp}	10	Number of reputation providers
Δ	10	Learning interval
λ	0.5	Success threshold
ψ	$[0, 1]$	Probability of leaving society

In our experiments, we have associated each profile with certain attributes and patterns. Using the proposed approach, the trustor tries to learn this association and exploits it to bootstrap trust. Table 2 lists the profiles used in our evaluations.

Table 2. Profiles and performance properties

Profile ID	Mean	SDV
P_1	0.9	0.05
P_2	0.6	0.15
P_3	0.4	0.15
P_4	0.3	0.1
P_5	0.0	1

Each trustee y has a set of attributes, each defined as a triple $\langle y, hasRel, i_{k:j} \rangle$ (shortly $hasRel(y, i_{k:j})$ hereafter) where $hasRel$ is the object property used to define attributes and $i_{k:j}$ is the j^{th} instance of a leaf concept C_k in the concept hierarchy. The hierarchy of concepts used in our simulations are shown in Figure 4.

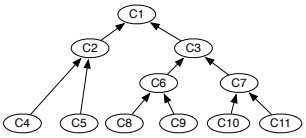

Fig. 4. Concept hierarchy used in simulations

We have 20 trustors and 100 agents (i.e., trustees) in our simulations. Each profile has equal number of trustees and each trustee can belong to one profile. Each simulation is run for 500 discrete time steps. At each step, each trustor selects one trustee to interact with. For this purpose, it gets the list of 10 randomly selected available trustees in the environment, evaluate their trustworthiness using a trust model and selects one with the highest trustworthiness. We have compared the following trust models in our evaluations.

- **Beta Reputation System (BRS):** Trust is estimated using Equation 2 where a is set to 0.5 [11].
- **Stereotypical Trust (ST):** Trust is estimated using Equation 2 where a is computed using a M5 regression model, which is trained using observable attributes of known agents and their trustworthiness [2].
- **Trust through Pattern Discovery (TPD):** Trust is estimated using the proposed approach.

The trustor also obtains a list of 10 reputation providers from the environment and queries them for evidence (i.e., past interaction outcomes) about trustees. In this work, other trustors serve as reputation providers and we assume they honestly share their evidence about trustees. This assumption is made only to focus on bootstrapping trust in this work. Handling deceptive evidence is out of the scope of this paper, but our approach be extended to handle deceptive evidence as described in [7].

To simulate the dynamical of the environment, we introduce the parameter ψ that determines the probability that a trustee leave the society. When a trustess leaves the society, a new trustee of the same profile joins the society. In this way, we maintain the balance of profiles and also simulate *whitewashing* behaviour of trustees.

In the following sections, we evaluate our approach against three settings: i) performance of trustees are not correlated with their descriptions, ii) their performances are correlated with their attributes, and iii) their performances are correlated with the patterns in their descriptions. We repeated our experiments 5 times and report their average results. As performance metric, we use the ratio of successful interactions. For frequent subgraph mining, we have used ParMol [14].

5.1 No Discriminative Features

In this setting, each trustee is assigned randomly a set of attributes. Hence, there is no correlation between participants' trustworthiness and their features. Figure 5 shows our results for this setting when no trustee leaves the society through out experiments ($\psi = 0.0$). In this setting, all of the three approaches have the same performance; they successfully determine the most trustworthy trustees in the environment and lead high ratio of successful interactions.

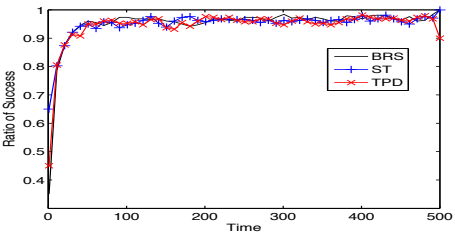

Fig. 5. Ratio of success when there is no discriminative features ($\psi = 0.0$)

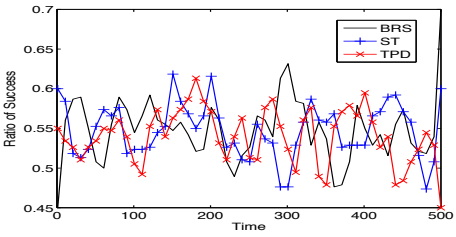

Fig. 6. Ratio of success when there is no discriminative features ($\psi = 0.5$)

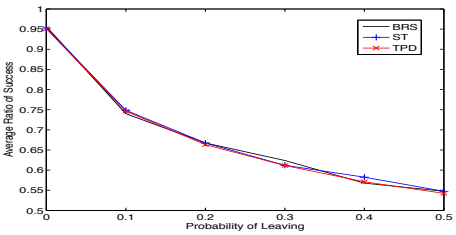

Fig. 7. Ratio of success as ψ changes when there is no discriminative features

Figure 6 shows our results for this setting when ψ is increased to 0.5. In this setting, trust approaches do not have enough interaction history to learn trustworthiness of each trustee. Hence, they have much lower performance compared to the case $\psi = 0$.

We repeated our experiments for different ψ values. Our results are shown in Figure 7. Our experiments indicate that the proposed approaches does not do worse than the existing approaches when there is no correlation between features of trustees and their performance. In such settings, the statistical trust approaches fail mostly because of the lack of enough evidence about the trustworthiness.

5.2 Attributes as Discriminative Features

In this setting, we associate the attributes in Table 8 to profiles as shown in Table 9. In this way, we set all trustees sharing the same profile have some common attributes in addition to their randomly assigned attributes.

Figure 10 shows our results for this setting for different values of ψ. The figure indicates that the performance of BRS significantly decreases as the probability of leaving the society increases. BRS assumes the same a priori trust for all trustees and build trustworthiness for individual trustees over time. However, as ψ increases, BRS could not have enough evidence to compute trust precisely. Unlike BRS, ST learns stereotypes about trustworthiness based on the attributes of trustees. Hence, it exploits the correlation between attributes of trustees and their attributes to precisely estimate a priori trust for trustees. As a result, ST have a very high success ratio that does not decrease with ψ. TPD successfully discovers the patterns implied by the attributes in Table 8 and uses these patterns as features to learn a mapping between features and a priori trust. As a result, in this setting, TPD is as successful as ST and correctly estimates trustworthiness even at high values of ψ.

a_1	$hasRel(?x, i_{4:1})$
a_2	$hasRel(?x, i_{5:1})$
a_3	$hasRel(?x, i_{8:1})$
a_4	$hasRel(?x, i_{9:1})$
a_5	$hasRel(?x, i_{10:1})$
a_6	$hasRel(?x, i_{11:1})$

Fig. 8. Attribute definitions

Profile ID	a_1	a_2	a_3	a_4	a_5	a_6
P_1	X					X
P_2		X		X		
P_3			X	X		
P_4		X	X		X	
P_5		X	X			X

Fig. 9. Profiles descriptions

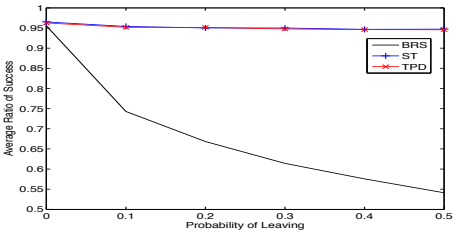

Fig. 10. Ratio of success as ψ changes when attributes are discriminative features

5.3 Patterns as Discriminative Features

In this setting, we associate the pattern in Table 3 to profiles as shown in Table 4. In this way, we set all trustees sharing the same profile have some common patterns.

In this challenging setting, trust approaches like BRS and ST could not estimate trustworthiness correctly as ψ increases. ST mainly fails because it could not find any mapping between attributes of trustees and their trustworthiness. On the other hand, the trustees belonging to the same profile share patterns. TPD discovers these patterns correctly and uses them to learn the correlation between these features and trustworthiness of trustees. Our results clearly show that TPD can estimate trustworthiness very successfully even in highly dynamic environments.

Table 3. Patterns and their definitions

p_0	$hasRel(?x, ?a) \wedge type(?a, C_1)$
p_1	$hasRel(?x, ?a) \wedge type(?a, C_2) \wedge hasRel(?x, ?b) \wedge type(?b, C_3)$
p_2	$hasRel(?x, ?a) \wedge type(?a, C_4) \wedge hasRel(?x, ?b) \wedge type(?b, C_6)$
p_3	$hasRel(?x, ?a) \wedge type(?a, C_5) \wedge hasRel(?x, ?b) \wedge type(?b, C_7)$
p_4	$hasRel(?x, ?a) \wedge hasRel(?a, ?b) \wedge hasRel(?b, ?c) \wedge type(?c, C_3)$
p_5	$hasRel(?x, ?a) \wedge hasRel(?a, ?b) \wedge hasRel(?b, ?c) \wedge type(?c, C_6)$
p_6	$hasRel(?x, ?a) \wedge hasRel(?a, ?b) \wedge hasRel(?b, ?c) \wedge type(?c, C_7)$

Table 4. Profiles described using patterns of Table 3

Profile ID	p_1	p_2	p_3	p_4	p_5	p_6
P_1	X	X				
P_2	X		X			
P_3				X		
P_4				X	X	
P_5				X		X

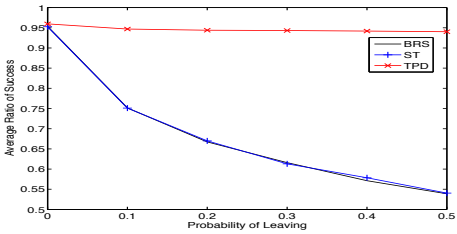

Fig. 11. Ratio of success as ψ changes when patterns are discriminative features

6 Discussion

There are a couple of statistical models for computing trust and reputation in multiagent systems. The beta reputation system (BRS) is proposed by Jøsang and Ismail [11]. It estimates reputations of service providers using Subjective Logic, where the trust is modelled using the beta probability density function. TRAVOS is proposed by Teacy *et al.* [17]. Similar to BRS, it uses beta probability density functions to compute consumers' trust on service providers. Caverlee *et al.* [4] propose the *SocialTrust* framework for tamper-resilient trust establishment in online social networks. In this framework, initially all users have the same level of trust. Then, SocialTrust dynamically revise trust ratings based on the interaction history.

The approaches mentioned above use direct or indirect past interactions with the other agents to compute trust. However when a new agent enters to a society, there is no direct or reputational evidence about the agent. Hence, bootstrapping trust becomes a challenge in open and dynamic multiagent systems. To address this challenge, Liu *et al.* propose agents to form stereotypes using their previous transactions with others [13]. In their approach, a stereotype contains certain observable attributes of agents and an expected outcome of the transaction. Similarly, Burnett *et al.* proposed to use stereotyping to bootstrap trust evaluations based on Subjective Logic [2]. Their approach allows agents to generalise their experience with known agents as stereotypes and apply these when evaluating new and unknown agents. Stereotypes are learned with standard *M5 regression tree* algorithm using a training set composed of observable attributes of known agents and their trustworthiness.

In this paper, we argue that agents with similar behaviour may share some patterns in their descriptions or relationships. We propose to discover these significant patterns in trustworthy and untrustworthy agents and exploit them to learn bootstrapping trust in dynamic and uncertain environments. Through extensive simulations, we show that the proposed approach is at least as good as the existing approaches in all settings. However, it significantly outperforms existing approaches when agents with similar behaviour share some common patterns. As a future work, we would like to use our approach to estimate trust in social networks with real data.

References

1. Alpaydin, E.: Introduction to Machine Learning. MIT Press (2001)
2. Burnett, C., Norman, T.J., Sycara, K.: Bootstrapping trust evaluations through stereotypes. In: Proceedings of Autonomous Agents and Multiagent Systems (AAMAS 2010), pp. 241–248 (2010)
3. Cao, L., Gorodetsky, V., Mitkas, P.: Agent mining: The synergy of agents and data mining. IEEE Intelligent Systems 24(3), 64–72 (2009)
4. Caverlee, J., Liu, L., Webb, S.: Towards robust trust establishment in web-based social networks with SocialTrust. In: WWW 2008: Proceeding of the 17th International Conference on World Wide Web, pp. 1163–1164 (2008)
5. Chakrabarti, D., Faloutsos, C.: Graph mining: Laws, generators, and algorithms. ACM Comput. Surv. 38 (June 2006)
6. Cook, D.J., Holder, L.B.: Graph-based data mining. IEEE Intelligent Systems 15(2), 32–41 (2000)
7. Şensoy, M., Zhang, J., Yolum, P., Cohen, R.: Poyraz: Context-aware service selection under deception. Computational Intelligence 25(4), 335–366 (2009)
8. Donato, D., Gionis, A.: A survey of graph mining for web applications. In: Aggarwal, C.C., Wang, H., Elmagarmid, A.K. (eds.) Managing and Mining Graph Data, pp. 455–485 (2010)
9. W.O.W. Group. OWL 2 web ontology language: Document overview (2009), http://www.w3.org/TR/owl2-overview
10. Jøsang, A.: Subjective Logic. Book Draft (2011)
11. Jøsang, A., Ismail, R.: The beta reputation system. In: Proceedings of the Fifteenth Bled Electronic Commerce Conference e-Reality: Constructing the e-Economy, pp. 48–64 (June 2002)
12. Jøsang, A., Ismail, R., Boyd, C.: A survey of trust and reputation systems for online service provision. Decis. Support Syst. 43, 618–644 (2007)
13. Liu, X., Datta, A., Rzadca, K., Lim, E.-P.: Stereotrust: a group based personalized trust model. In: Proceeding of the ACM Conference on Information and Knowledge Management, pp. 7–16 (2009)
14. Meinl, T., Wörlein, M., Urzova, O., Fischer, I., Philippsen, M.: The ParMol package for frequent subgraph mining. In: Electronic Communications of the EASST, pp. 94–105 (2007)
15. Quinlan, J.: Learning with continuous classes. In: Proceedings of the 5th Australian Joint Conference on Artificial Intelligence, Singapore, pp. 343–348 (1992)
16. Sirin, E., Parsia, B., Grau, B.C., Kalyanpur, A., Katz, Y.: Pellet: A practical OWL-DL reasoner. Web Semant. 5(2), 51–53 (2007)
17. Teacy, W., Patel, J., Jennings, N., Luck, M.: TRAVOS: Trust and reputation in the context of inaccurate information sources. Autonomous Agents and Multi-Agent Systems 12(2), 183–198 (2006)
18. Yan, X., Han, J.: gSpan: Graph-based substructure pattern mining. In: Proceedings of the International Conference on Data Mining, pp. 721–724 (2002)