

Positional Reputation Systems

Marco Voss, Lars Brückner, and Andreas Schlosser

Department of Computer Science
Darmstadt University of Technology
Hochschulstraße 10
D-64289 Darmstadt, Germany
`{voss,brueckner,schlossr}@ito.tu-darmstadt.de`

Abstract. We present a new approach for computing reputation in agent societies. Our positional reputation system can handle arbitrary dimensions to provide a mapping of different attributes of an agent to its reputation value. The meaning of a certain dimension is not fixed, but is defined dynamically by the agent community. Thus a positional reputation system automatically adapts to the needs of the using community. The computed reputations by such a positional reputation system are local, i.e. they depend on the point of view of the requesting agent.

1 Introduction

Whenever mutually unknown agents meet each other to engage in transactions, serve as routers in mobile ad-hoc networks or talk over arbitrary topics in discussion boards, reputation systems may serve as a mechanism for building trust between these agents. Our informally defined understanding of the terms *reputation* and *trust* is:

Reputation is the collected and processed information about one entity's former behavior as experienced by others.

"*Trust* is a measure of willingness to proceed with an action (decision) which places parties (entities) at risk of harm and is based on an assessment of the risks, rewards and reputations associated with all the parties involved in a given situation." [1]

A reputation system aggregates the ratings it has collected to compute a reputation value for each participating agent [2]. This reputation value can be the basis for an agent's trust decision. Most classical systems accept only a one-dimensional rating for an encounter between agents [3–7], i.e. a rating like *good* or *bad* or a rating from an interval like $[-1, \dots, 1]$. The meaning of this rating derives from the context of the transaction. A positive rating collected by *eBay* [8] characterizes an agent as a reliable seller or buyer, whereas a rating received in a routing context [9] characterizes a node as a reliable router. By this implicit definition of the context and meaning of a reputation value these systems are restricted to specific applications. These classical reputation systems have their right to exist through the applications mentioned above.

However, there are application scenarios for reputation systems where one-dimensional ratings and reputations are inadequate. eBay for instance, mingles the two different roles of buyer and seller and ignores the value of a rated transaction. This limits the expressiveness of the reputation system. Another example are discussion boards that often treat very different topics. Here, the same agent may be an expert in one topic $T_{\text{mathematics}}$ and a novice in another topic $T_{\text{frenchcooking}}$. If the ratings for his posts were aggregated into a one-dimensional reputation value his expertise in topic $T_{\text{mathematics}}$ would be influenced negatively by bad ratings he received for posts in topic $T_{\text{frenchcooking}}$.

Thus reputation systems supporting different viewpoints are needed. The approach presented in this paper does not fix each dimension of a reputation resp. rating to a certain attribute, but allows the community to define the meaning *dynamically*. There are no *good* or *bad* ratings, but ratings that manipulate an agent's *position*, e.g. a reputation vector. The reputation of an agent from another agent's point of view is determined by the distance between both's positions. The more dimensions the system provides the more detailed may the different attributes be decomposed.

The intention of this paper is not to present a complete solution but to formulate a new concept of reputation that emphasizes an agent's own position. The remainder of the paper is organized as follows. Section 2 gives an abstract model for positional reputation systems and introduces the used terminology. We define requirements for positional reputation systems in Sec. 3. The idea of positional reputation systems is compared to related approaches in Sec. 4. An example metric using a positional reputation system is presented in Sec. 5 and a possible architecture for such a reputation system in Sec. 6. Possible future work is drafted in Sec. 7 and the paper is concluded in Sec. 8.

2 A Model for Positional Reputation Systems

We now present an abstract model for reputation systems. In contrast to other models we distinguish between an agent's *position* as a resume of the ratings given to the agent in the past, and the agent's *reputation* as an interpretation of the agent's position in relation to the current peer's own position. *Ratings*, which are a manifestation of an agent's opinion about the last transaction, are transformed into *position changes*. Within our model, the application runs through three steps: trust building, transaction, and rating.

The first step, trust building, is the process of deciding whether to start a transaction or not. The agent checks the reputation value of the potential transaction partner, but other factors like the value of the transaction are part of the decision process, too.

Every agent a has a *position* that represents the agent's status within the community S at certain time t :

$$pos_a(t) \in S$$

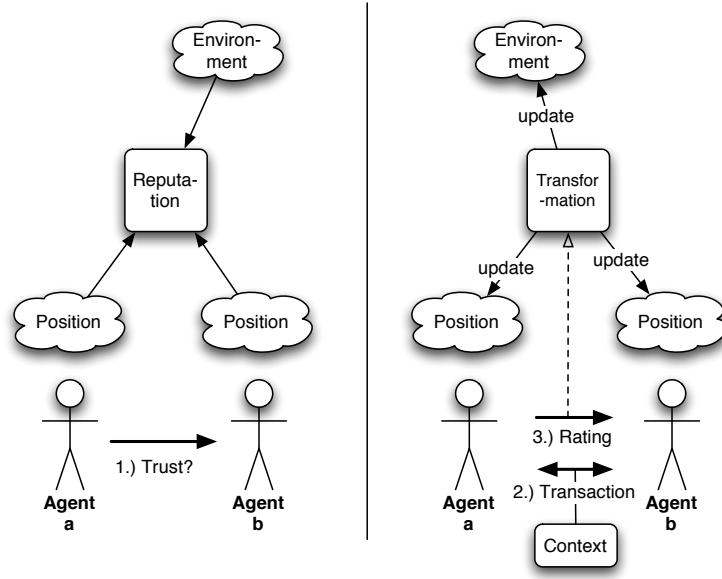


Fig. 1. Positional Reputation Model

The actual *reputation rep* of an agent a depends on the positions of a and the requesting agent b , and the environment E at a given time t , like depicted on the lefthand side of Fig. 1:

$$rep : (S \times S \times E) \rightarrow R$$

The environment may include other agents' positions, e.g. recognized clusters, and general state information of the reputation system. What elements are part of the environment and their impact on the reputation value is defined by the reputation metric. The decision process whether to trust the other peer is not further discussed here. If the agents trust each other, they proceed with the transaction phase. The rating phase begins after the transaction is complete. Based on the outcome of the transaction, b concludes what rating v it wants to give to a . The rating value $v \in V$ expresses b 's opinion on the last transaction and its partner b . Depending on the application, V may be an interval like $[-1, \dots, 1]$ or a fixed set of labels like $\{very\ good, good, bad, off\text{-}topic, offensive\}$.

A set of transformation functions updates the positions of a , b and the environment to incorporate the rating, like illustrated on the righthand side of Fig. 1:

$$T_{subject} : (S \times V \times E) \rightarrow S$$

$$T_{rater} : (S \times V \times E) \rightarrow S$$

$$T_{env} : (E \times V) \rightarrow E$$

Existing reputation systems can be easily mapped to this extended model. Additive systems like eBay, for instance, have a one-dimensional scale with positive orientation: $S := \mathbb{N}$. The reputation function rep is the identity function and ignores the requesters position and the environment. Therefore, the transformations T_{rater} and T_{env} are not used.

$$T_{subject} : (x, v, e) \mapsto \begin{cases} x + 1 & \text{if } v = \text{"positive"} \\ x & \text{if } v = \text{"neutral"} \\ x - 1 & \text{if } v = \text{"negative"} \end{cases}$$

where $x \in \mathbb{N}$, $v \in \{\text{"positive"}, \text{"neutral"}, \text{"negative"}\}$ and $e \in E$.

3 Positional Reputation Systems

The distinction of position and reputation allows us to define the following properties as requirements for a *positional reputation system*:

- Users with similiar resumes should have a high reputation when they interact with each other.
- No part of the position value should be defined to map directly to a certain real-world property of the user.
- A position change value should not reveal to the user how many ratings were issued, which peers have issued the ratings and how the other agents rated him.
- A rating process may change both the rater’s and the ratee’s position value.
- The reputation metric must prevent that an adversary earns a high reputation just by giving ratings.
- The transformation functions T should prevent that the related ratings nullify each other.
- If an agent manages to skip (negative) position changes, he should still be penalized by the position changes of his raters and the updates to the environment.

The above requirements make certain attacks and abusive practices harder to implement because there is no clear target. It does not make sense to drop ratings intentionally, if one cannot distinguish between good or bad ratings. Equally, one cannot prevent that other agents move away from oneself by changing their own position. The so called ballot stuffing attack, where a group of colluding agents try to pushup each others reputation, does not work when there is no predefined orientation. If they try to push an agent into the direction of a reputable position they cannot avert that other agents reinterpret this position. This may lead to a devaluation.

4 Related work

Collaborative filtering systems like the MovieLens [10] movie recommender system were a major inspiration for the formal model in Sec. 2. Within MovieLens,

each user has a profile that contains the user's own ratings on different movies. The system compares all profiles to find users who have given similar ratings to the same movies. The system recommends those movies to the user that other users with a similar profile have rated positive but that are not rated within his own profile yet. The difference to our work is that we do not assume that any element of the position has a semantic meaning; within MovieLens, every element of the profile stands for the user's opinion about a specific movie. Also, within a recommender system, a user's rating only changes the user's own profile. Speaking in terms of our model, a recommender system allows the user to choose his own position without any restrictions. Trying to harm the usefulness of the recommender system by giving senseless ratings is countered by the mass of honest users. Within most reputation systems, a rating is applied to another user's profile. A reputation system is about recommending another user himself as a transaction partner to others in the future. In a positional reputation system, a user's rating changes both profiles. The reputation metric has to put a limit on the user's possibilities to change his own or other's profiles; otherwise the reputation might become too unstable to provide useful reputation values.

Research on information retrieval deals with the similar problem of finding (text) documents matching a specific query. An important information retrieval technique is based on a vector space model first implemented in the SMART system [11, 12]. Index terms are used as basis vectors in a linear vector space. Documents and queries are represented as vectors according to the occurrence of index terms. A similarity function is used to create a ranking of documents with respect to a query. Common similarity functions are the scalar product or the cosine coefficient. Advanced vector space systems adapt the weighting factors for the query vector and documents to achieve better results. Our geometrical model presented in Sec. 5 has some relationship to this kind of systems, whereby our approach changes the vectors dynamically with every transaction.

A lot of local reputation systems have been proposed, which provide a personalized computation of reputation to each agent [13, 14]. But all these systems rely on a fixed scale for their reputation values, i.e. each rating has a fixed meaning while our approach allows the community to define the meaning of a rating dynamically. Local reputation systems are often used in the context of mobile ad-hoc networks and have to take care of the data dissemination between agents [15]. With a positional reputation system each agent could store his own position. Because a single rating has no meaning, an agent cannot decide for a single rating if it is a good or bad one and thus has no incentive to discard received ratings. Only the position itself has to be protected by some dependable security mechanisms to prevent direct manipulation of the position.

A static approach to introduce several dimensions into reputation systems is done by ontologies [16, 17]. Ratings are given according to a certain attribute. The relationship between these attributes is fixed by the system through defining the ontology. The quality of such a system depends heavily on the quality of the underlying ontology. The ontology has to be defined when the system is deployed. Changes to the ontology at a later time are often problematic.

5 An Example for a Positional Reputation Metric

The above definition of a positional reputation system can be interpreted geometrically. Thereby, the terms *position*, *distance* and *movement* get an implicit meaning. Nevertheless, we want to point out that the general model should not be restricted to this interpretation. It should be part of future research to develop other (i.e. non-geometric) kinds of systems and metrics. However, the geometrical model allows to "illustrate" the abstract definitions and poses already a lot of interesting research questions: What mathematical properties should the underlying space (or better manifold) have? Which distance measures are to be used? What kind of movement strategies should be applied?

A simple geometrical implementation of a positional reputation system may for instance use an n -dimensional torus as underlying space. The position v of an agent is represented by a point on this torus: $v \in \mathbb{T}^n$. \mathbb{T}^n can be constructed by taking a Cartesian product of n copies of the unit sphere \mathbb{S}^1 :

$$\mathbb{T}^n := (\mathbb{S}^1)^n$$

A 2-dimensional instance of a ring torus can be constructed from a rectangle by gluing both pairs of opposite edges together with no twists. It has the shape of a "donut". The distance function is an adapted euclidian metric

$$d(v, w) := \sqrt{\sum_{i=1}^2 (v_i - w_i)^2}$$

where $v := (v_1, v_2) \in \mathbb{T}^2$ and $w := (w_1, w_2) \in \mathbb{T}^2$ are position vectors of two different agents and the "-" operator takes the wrapping into account.

Figure 2 shows a simulation of this 2-dimensional scenario implemented with the Repast [18] agent simulation toolkit. At the beginning agents are placed randomly on the torus. An agent has a taste, which is illustrated by a certain color. Every round a percentage of agents to be active next is selected. If an agent sees an acting agent of similar taste it moves towards its direction and tries to poll it nearer, i.e. it manipulates the positions by a "positive" rating to alleviate the distance between them. Otherwise it moves away and pushes the other agent in the opposite direction, as depicted by the arrows in the left image. The different impulses an active agent receives from other agents are summed up to a resulting movement, i.e. a change of this agent's position by the transformation functions T_x induced by the received ratings. An agent has a limited horizon (illustrated by the circle in Fig. 2(a)) meaning that it only recognizes actions inside this radius. After some time agents cluster to groups of similar taste.

The environment E can contain information about the cluster an agent belongs to. For example, if there are two agents located in the same distance from the asking agent but belonging to different clusters this information can help to make a distinction. Figure 2(b) shows the clustering that came up through the given ratings.

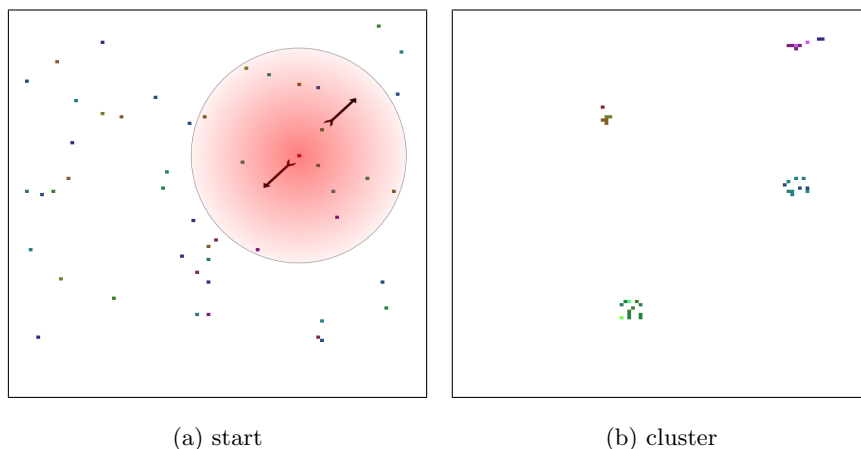


Fig. 2. Simulation

Our first implementation of the simulation is very basic. Currently it is more suitable for illustration purposes than to provide an in deep investigation. It is not yet able to deal with attacks or problems like local clustering or how to choose and adapt the horizon. It is be part of the future work to implement an experimentation platform that takes these issues into account and supports different geometries.

Other geometrical models may use different surfaces and distance functions. An example is the surface of a n -dimensional unit sphere S^n , where not only the projected position on this sphere but also the height is evaluated. The distance on the unit sphere can be interpreted as difference in taste, and the length of a position vector as the expertise of an agent.

6 An Implementation Scenario

This section describes a draft for the implementation of a positional reputation system for web blogs. The system presented here uses local storage of the positions and allows the creation of reputation realms across independent blogs hosted by different parties. A key feature of web blogs, like discussion forums and wikis, is the possibility that any user may create new content, which may include links to other web sites, and add it to the website. Blogs suffer the same problems as internet news and e-mail: spreading of false information, defamation, spamming by advertisers, and rude behavior against people with different opinions. Therefore, a lot of blogs require the users to register themselves and supply an e-mail address before they can post. Larger discussion forums rely on moderators to remove offensive or off-topic posts. We propose a positional reputation system to keep abusive posters and spammers at bay.

Within our system, the user needs a special reputation client software installed on his computer. It may be implemented as a browser extension or as an http proxy. The reputation client stores the user's position and sends it along to the blog host when required. The blog servers implement the reputation function and the transformation for the positions. Furthermore, blogs with similar discussion topics form a web of trust (e.g. by using PGP) to allow the usage of the same position across multiple sites.

Whenever a user publishes a new article, his reputation client passes the user's current position along. The blog server stores a local copy of the position. For every rating the article receives, the blog server calculates the resulting position change and applies it to his local copy. After a certain time the author's client software contacts the blog server again to receive the overall position update.

Whenever a user queries the blog server for new articles, the reputation client passes the user's current position along with the query. The blog server calculates the reputation values for all articles and sends the results back to the client. If the user submits ratings, the blog server calculates the position update for the user and the author.

These remarks show the basic architecture of an application of positional reputation systems. Further discussion of implementation issues is out of scope of this paper.

7 Future Work

We see three major areas for future work on this subject: research on reputation metrics, applicability for one-to-one transactions, and privacy issues.

The positional metric presented in sec. 5 separates different groups of agents apart from each other, but it does not provide a mean to establish a hierarchy inside a peer group, i.e. to identify experts. Earlier results from the field of information retrieval will surely be helpful to find better metrics.

We presented web blogs as one main usage scenario for positional reputation systems. Within that scenario, all transactions are 1:n, meaning that one author gets n ratings per article. To increase privacy and protect users from defamation, we have introduced the positional reputation system as an alternative to a traditional authentication and rating scheme. Further research is necessary to investigate whether other applications, especially those with 1:1 transactions can benefit as well from a positional reputation system.

In sec. 6, we proposed to use a positional reputation system whose positions could be shared across different servers. The implementation of such a system needs to address a lot of security and privacy issues. Users should not be able to freely choose their own position or hijack another user's position. An adversary might start his own blog server and collect all positions sent by the users. By correlating single positions with information from the articles that a user has authored or rated, the adversary could try to extract personal information from position values and apply data mining techniques to find groups of similar users.

The challenge is to find a metric whose position provides enough information for the function of the reputation system but that is resistant enough against this clustering attack. Another point is the linkability of different articles or ratings from the same user, which may not be desirable in some scenarios.

8 Conclusions

In this paper we presented a new approach for reputation systems, the positional reputation system. The key features are the distinction between positions and reputations, and the dynamic semantics of ratings. Many known attacks to reputation systems like ballot stuffing are hard to arrange within positional reputation systems. A simple 2-dimensional positional reputation system was implemented as a proof of concept, and we observed the desired clustering of agents with similar taste. We identified web blogs as a main application area for our approach, since this domain requires a reputation system which is able to map many different attributes into a reputation value. These attributes may develop dynamically during the lifetime of the blog and our system is able to adapt to these changes. Furthermore, we gave a sketch of an architecture of web blogs using positional reputation systems.

References

1. Mahoney, G.: Trust, Distributed Systems, and the Sybil Attack (2002)
2. Resnick, P., Zeckhauser, R.: Reputation Systems. *Communications of the ACM* **43** (2000) 45–48
3. Jøsang, A., Ismail, R.: The Beta Reputation System. In: Proceedings of the 15th Bled Conference on Electronic Commerce, Bled, Slovenia, 17-19 June 2002. (2002)
4. Kamvar, S.D., Schlosser, M.T., Garcia-Molina, H.: The Eigentrust Algorithm for Reputation Management in P2P Networks. In: Proceedings of the twelfth international conference on World Wide Web, ACM Press (2003) 640–651
5. Selçuk, A.A., Uzun, E., Pariente, M.R.: A Reputation-based Trust Management System for P2P Networks. In: Proceedings of the 4th IEEE/ACM International Symposium on Cluster Computing and the Grid (CCGrid04). (2004)
6. Sabater, J., Sierra, C.: REGRET: A Reputation Model for Gregarious Societies. In: Proceedings of the Fifth International Conference on Autonomous Agents. (2001)
7. Buchegger, S., Le Boudec, J.Y.: A Robust Reputation System for Mobile Ad-hoc Networks. Technical Report, EPFL, Switzerland (2003)
8. N.N.: ebay Homepage (2005) <http://www.ebay.com>.
9. Yau, P., Mitchell, C.J.: Reputation Methods for Routing Security for Mobile ad hoc Networks. In: Proc. of SympoTIC '03, Joint IST Workshop on Mobile Future and Symposium on Trends in Communications, Bratislava, IEEE Press (2003)
10. N.N.: Movielens (2005) <http://www.movielens.org>.
11. Salton, G., ed.: The SMART retrieval system. Prentice Hall, Englewood Cliffs (1971)
12. Wong, S.K.M., Ziarko, W., Wong, P.C.N.: Generalized vector spaces model in information retrieval. In: SIGIR '85: Proceedings of the 8th annual international ACM SIGIR conference on Research and development in information retrieval, ACM Press (1985) 18–25

13. Zacharia, G., Maes, P.: Trust Management through Reputation Mechanisms. *Applied Artificial Intelligence* 14 (2000)
14. Buchegger, S., Le Boudec, J.Y.: The Effect of Rumor Spreading in Reputation Systems for Mobile Ad-hoc Networks. In: Proc. WiOpt'03 (Modeling and Optimization in Mobile, Ad Hoc and Wireless Networks). (2003)
15. Heidemann, J., Silva, F., Estrin, D.: Matching Data Dissemination Algorithms to Application Requirements. In: Proceedings of the ACM SenSys Conference. (2003)
16. Rezgui, A., Bouguettaya, A., Malik, Z.: A Reputation-Based Approach to Preserving Privacy in Web Services. In: Lecture Notes in Computer Science. Volume 2819., Springer (2003)
17. Maximilien, E.M., Singh, M.P.: An Ontology for Web Service Ratings and Reputations. In Cranefield, S., Finin, T.W., Tamma, V.A.M., Willmott, S., eds.: Proceedings of the Workshop on Ontologies in Agent Systems (OAS 2003) at the 2nd International Joint Conference on Autonomous Agents and Multi-Agent Systems, Melbourne, Australia, July 15, 2003. Volume 73 of CEUR Workshop Proceedings. (2003)
18. RePast Homepage: (2005) <http://repast.sourceforge.net>.