Electronic Preprint for Journal of Information Processing Vol.24 No.2

Regular Paper

A Reputation System that Resists a Collusive Attack Focused on a Specific Target

Fumiaki Sato[1,a)]

Received: May 14, 2015, Accepted: November 5, 2015

Abstract: The rapid growth of communication networks, such as the Internet and mobile networks, has spurred the development of numerous online trading systems. It is important for such a community that there be trust between the trading partners. A system that stores information about the reputation for trustworthiness of the individuals in the network is known as a *reputation system*. However, a malicious evaluator can provide an incorrect evaluation, and several such users may collude to create a false evaluation of a particular target; this is known as a pinpoint collusive attack and is a major problem. In order to resist such an attack, reputation systems have been proposed that consider the ability of a given evaluator to provide an accurate evaluation of a given target. However, these systems are unable to identify attacks from evaluators who have achieved a sufficiently high rating for their ability. In this paper, we propose a novel algorithm that, when calculating the ability of an evaluator, reduces the influence of values that are not in line with the daily trend. Simulation results show that the proposed method reduces the influence of an attack.

Keywords: reputation systems, collusive attack, online trust

1. Introduction

The increase in the connectivity of the Internet has resulted in a rapid increase in the popularity of online communities. In addition, Internet marketplaces (such as eBay.com), popular P2P protocols (such as BitTorrent), and Web 2.0 applications (such as Facebook and YouTube) attract users worldwide by offering novel content and services.

While these next-generation Internet communities offer a variety of opportunities, there is also a risk involved for their members. These applications rely primarily on cooperative user behavior for their correct operation. Moreover, the field of applications is open and anonymous, which makes online communities vulnerable to attacks from malicious and self-interested members.

In order to reduce such transaction risks and improve performance, applications must manage the trust between users, and a reputation system seems to be at the heart of all online transactional activities that require trust [1], [2], [3], [4].

In recent years, there have been many studies of reputation systems. These have considered wide-ranging applications, such as P2P systems [5], [6], ad hoc networks [7], [8], wireless sensor networks [9], [10], and spam-filtering systems [11]. In this paper, we will discuss a reputation system for an information-sharing system in an online community.

The simplest reputation system relies on averaging the evaluations of each user. However, since the ability of each evaluator to provide an accurate evaluation is different, a more sophisticated approach is to compute a weighted average, in which each evaluation is weighted in accordance with the reliability of the evaluator; this reduces the influence of a false or malicious evaluation. However, there are difficulties in calculating the ability (potential to perform well) or reliability (history of performing well) of the evaluators, since this depends on the support that they have received from other users and on their previous responses to other users. To complicate this, prior to an attack, users may collude to illegally enhance the ability and reliability of one or more evaluators. In particular, a "Sybil attack" may occur when a user obtains multiple IDs, which are then used to illegally enhance their own apparent ability and reliability.

To address such attacks, one approach [1] is to include in the evaluation of the reliability, the size of the deviation of the reliability from that of other users; this is based on the idea that a large deviation in reliability may indicate a collusion attack. However, the reliability of evaluator is not considered.

EigenRumor [12], [13] is an algorithm for reputation systems for information-sharing in an online community. There are other algorithms for such systems, including PageRank [14]), but EigenRumor uses an agent's score instead of the link's score, and thus it is resistant to a Sybil attack. In EigenRumor, the shared information is evaluated by the members of the community, and the reputation of the contributor is determined using that evaluation combined with the ability of the evaluator. Therefore, EigenRumor is resistant to a Sybil attack by a member with low ability, but it is not resistant to attacks by members of sufficiently high ability, even if this was gained by illegal means. In this paper, we propose an algorithm that prevents collusive attacks by disabling any evaluation that is different from the trend of that evaluator; this is, it considers that evaluator's history when de-

[1] Department of Information Science, Toho University, Funabashi, Chiba 274–8510, Japan
[a)] fsato@is.sci.toho-u.ac.jp

termining whether a particular evaluation should be accepted. In addition, we perform a simulation to show the effectiveness of the proposed method.

The remainder of this paper is organized as follows. We begin by presenting an overview of the EigenRumor algorithm, in Section 2. In Section 3, we propose our algorithm for enhancing the EigenRumor algorithm so that it can resist a pinpoint collusive attack. Simulation results are presented and discussed in Section 4. Finally, we present our conclusions in Section 5.

2. EigenRumor

2.1 Community Model

The community model of EigenRumor assumes m agents and n information objects. When agent i provides object j, a provisioning link is established from i to j. The provisioning matrix $P = [p_{ij}]$ ($i = 1 \ldots m$, $j = 1 \ldots n$) represents all provisioning links in the universe. In this matrix, $p_{ij} = 1$ if agent i provides object j, and it is zero otherwise. When agent i evaluates the usefulness of an existing object j with scoring value e_{ij}, an evaluation link is established from i to j. The evaluation matrix $E = [e_{ij}]$ ($i = 1 \ldots m$, $j = 1 \ldots n$) represents all evaluation links in the universe. An evaluation link is assigned weight e_{ij} based on the strength of the support given to object j; e_{ij} is in the range [0, 1], and higher values indicate stronger support. **Figure 1** shows the EigenRumor community model.

2.2 The EigenRumor Algorithm

The EigenRumor algorithm calculates three vectors: the **authority vector** $\vec{a}$, the **hub vector** $\vec{h}$, and the **reputation vector** $\vec{r}$. The authority vector contains the **authority score**, which indicates the ability of the agent to provide information objects to the community; see Eq. (3). The hub vector contains the **hub score**, which indicates the ability of the agent to contribute accurate evaluations to the community; see Eq. (4). The reputation vector contains the **reputation score**, which indicates the level of support that the object received from the community; see Eq. (5).

These vectors are based on the following assumptions.

Assumption 1: Objects that are provided by a "good" authority will follow the trend of the community.

Assumption 2: Objects that are supported by a "good" hub will follow the trend of the community.

Assumption 3: Agents that provide objects that follow the community trend are "good" authorities of the community.

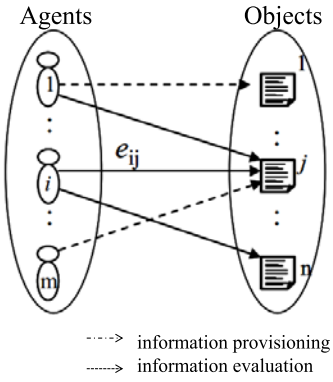


Fig. 1 EigenRumor community model.

Assumption 4: The agents that provide evaluations that follow the community trend are "good" hubs of the community.

Corresponding to the above assumptions, the algorithm contains the following four equations.

$$\vec{r} = P^T \vec{a} \tag{1}$$

$$\vec{r} = E^T \vec{h} \tag{2}$$

$$\vec{a} = P\vec{r} \tag{3}$$

$$\vec{h} = E\vec{r} \tag{4}$$

In order to merge Eqs. (1) and (2), the following equation is used.

$$\vec{r} = \alpha P^T \vec{a} + (1 - \alpha) E^T \vec{h} \tag{5}$$

Here, α is a constant in the range [0, 1], and it controls the weight given to the authority and hub scores. **Figure 2** shows the procedure used to calculate the equilibrium values for the score vectors.

2.3 Evaluation and Problem of the EigenRumor Algorithm

In order to evaluate the performance of the EigenRumor algorithm, we compare it to calculating the reputation score as a simple average of the evaluation values.

We assume there to be two types of the agents: those with high ability to evaluate the usefulness of the object and those with low ability. Each object has an intrinsic usefulness that is related to the number of the object; for example, the first object has little usefulness, and the tenth object is very useful. We assume that when agents with low ability evaluate an object, they assign it a number that does not reflect its true value (a "false evaluation"). We will assume there that are ten each of agents and objects, with two of the agents having low ability, and that there are 100 links to be evaluated.

Figure 3 shows the results of the calculation. For an object

$$\vec{a}^{(0)} = (1,\ldots,1)^T$$
$$\vec{h}^{(0)} = (1,\ldots,1)^T$$
while $\vec{r}$ *changes significantly do*
$$\vec{r}^{(k)} = \alpha P^T \vec{a}^{(k)} + (1-\alpha) E^T \vec{h}^{(k)}$$
$$\vec{r}^{(k+1)} = \vec{r}^{(k)} / \left\| \vec{r}^{(k)} \right\|_2$$
$$\vec{a}^{(k+1)} = P\vec{r}^{(k+1)}$$
$$\vec{h}^{(k+1)} = E\vec{r}^{(k+1)}$$
end while

Fig. 2 EigenRumor community model ($\|\bullet\|_2$ is the notation for the function that computes the L_2 vector norm).

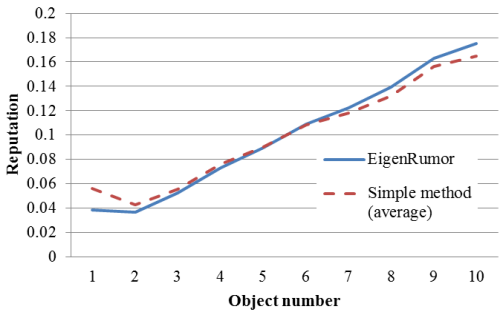


Fig. 3 Evaluation of the performance of EigenRumor.

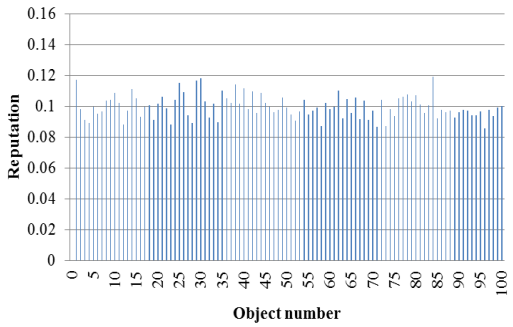


Fig. 4 Evaluation of the performance of EigenRumor.

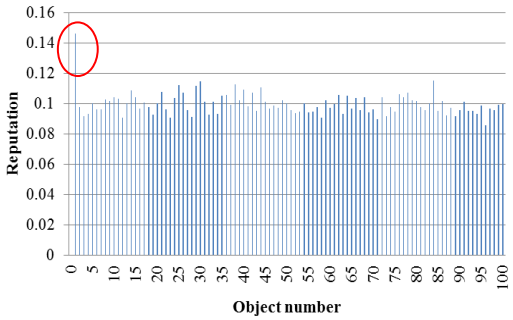


Fig. 5 Results of a simulation using EigenRumor with attack (10% of the agents were malicious).

$$\vec{a}^{(0)} = (1,...,1)^T$$
$$\vec{h}^{(0)} = (1,...,1)^T$$
$$Ave[j] = \frac{1}{m}\sum_{i=1}^{m} e_{ij}$$
$$DiffAve[i] = \frac{1}{n}\sum_{j=1}^{n} (e_{ij} - Ave[j])$$
$$\sigma[i] = \sqrt{\frac{1}{n}\left(\sum_{j=1}^{n} (e_{ij} - DiffAve[i])^2\right)}$$
$$Th[i] = \tau \cdot \sigma[i]$$
$$E' = [e'_{ij}]\ (e'_{ij} = e_{ij}, \quad if\ \left|e_{ij} - Ave[j]\right| \le Th[i],$$
$$e'_{ij} = Ave[j], \quad if\ \left|e_{ij} - Ave[j]\right| > Th[i])$$
while $\vec{r}$ *changes significantly do*
$$\vec{r}^{(k)} = \alpha P^T \vec{a}^{(k)} + (1-\alpha) E'^T \vec{h}^{(k)}$$
$$\vec{r}^{(k+1)} = \vec{r}^{(k)} / \left\|\vec{r}^{(k)}\right\|_2$$
$$\vec{a}^{(k+1)} = P\vec{r}^{(k+1)}$$
$$\vec{h}^{(k+1)} = E\vec{r}^{(k+1)}$$
end while

Fig. 6 Proposed Method (1).

with a smaller evaluation number, EigenRumor calculates a lower reputation, and for an object with a larger number, it calculates a higher reputation. In this evaluation, EigenRumor performed better than did the simple algorithm.

However, if agents with similar behavior join together to attack a particular target object, EigenRumor calculates a false value for the reputation. In order to evaluate this, we conducted a simulation, as follows. There were 100 each of agents and objects. Each object was assumed to have the same level of usefulness. Further, it was assumed that 10% of the agents were malicious, and they would give a large evaluation value to Object 1, and the correct value to the other objects. There were 10,000 links to be evaluated.

Figure 4 shows the simulation results when there were no malicious agents, and **Fig. 5** shows the simulation results when 10% of the agents were malicious. From these results, we see that the reputations calculated by the EigenRumor incorporate those of malicious agents who attack a single target object.

3. Proposal to Resist a Collusive Attack

3.1 Concept of the Proposal

In this section, we consider an attacker model in which malicious agents behave normally except for the evaluations that they give to a single specific target. By "normal," we mean that they give honest evaluations. Therefore, the hub scores of the malicious agents become large. We assume that multiple malicious agents collude to attack the same target and that they give an overly large value to the target in order to increase its reputation score. An intentionally incorrect evaluation is called a "false evaluation" or "malicious evaluation." This attack will increase the reputation score of the target object. The purpose of our method is to make the value of the reputation as close as possible to the value that it would have without the evaluation of any malicious agents.

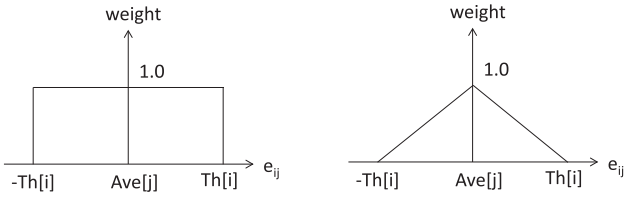
In this proposal, we will assume that the malicious agents only give a false evaluation to a single target object. If an agent gives an evaluation that is very different from the community trend, we will decide that it should not be included in the reputation score. The threshold to invalidate an evaluation is determined by the previous differences in the evaluations given by that community.

In order to increase this threshold, it is necessary to inflate the evaluations given to objects other than the target. However, in this case, EigenRumor will reduce the hub score (ability) of this agent, and thus the attempt to raise the threshold will not succeed.

Our proposed method may fail to detect a small attack that does not exceed the threshold value, but the effect of this will be small. In order to resist such an attack, we propose an enhanced method that, instead of using a threshold, assigns a weight that is based on the difference between a given evaluation and the current trend in the community.

3.2 Algorithm of Proposed Method (1)

The process for calculating the reputation is nearly the same as the method used in EigenRumor. However, if a given evaluation is significantly different from previous ones, we will replace it with to the community's average value when calculating its reputation using Eq. (5). **Figure 6** shows the calculation process of the scheme. $Ave[j]$ is the average value of the evaluations of object j, $DiffAve[i]$ is the average of the differences between each of the evaluations and the community trend for agent i, $\sigma[i]$ is the standard deviation of the differences between each of the evaluations and the community trend for agent i, and $Th[i] = \tau\sigma[i]$ is the threshold for agent i for replacing an evaluation with the community average. τ is a constant value.



(a) Weight assignment of method 1.

(b) Weight assignment of method 2.

Fig. 7 Weight assignments of proposed methods.

$$\vec{a}^{(0)} = (1,...,1)^T$$
$$\vec{h}^{(0)} = (1,...,1)^T$$
$$Ave[j] = \frac{1}{m}\sum_{i=1}^{m} e_{ij}$$
$$DiffAve[i] = \frac{1}{n}\sum_{j=1}^{n}(e_{ij} - Ave[j])$$
$$\sigma[i] = \sqrt{\frac{1}{n}\left(\sum_{j=1}^{n}(e_{ij} - DiffAve[i])^2\right)}$$
$$Th[i] = \tau \cdot \sigma[i]$$
$$E'' = [e''_{ij}]\ (e''_{ij} = Ave[j] + (e_{ij} - Ave[j])(1 - \frac{|e - Ave[j]|}{Th[i]}),$$
$$if\ |e_{ij} - Ave[j]| \le Th[i],$$
$$e''_{ij} = Ave[j], \qquad if\ |e_{ij} - Ave[j]| > Th[i])$$
while $\vec{r}$ *changes significantly do*
$$\vec{r}^{(k)} = \alpha P^T \vec{a}^{(k)} + (1-\alpha)E''^T \vec{h}^{(k)}$$
$$\vec{r}^{(k+1)} = \vec{r}^{(k)} / \left\|\vec{r}^{(k)}\right\|_2$$
$$\vec{a}^{(k+1)} = P\vec{r}^{(k+1)}$$
$$\vec{h}^{(k+1)} = E''\vec{r}^{(k+1)}$$
end while

Fig. 8 The proposed method 2.

In Fig. 6, E' is the new matrix of evaluations. An evaluation e'_{ij} of E' is equal to the original e_{ij} if $|e_{ij} - Ave[j]| \le Th[i]$. If $|e_{ij} - Ave[j]| > Th[i]$, then e'_{ij} is set equal to $Ave[j]$. Therefore, any evaluation that is sufficiently higher than the usual evaluation will be forced to the average value.

3.3 Algorithm of Proposed Method (2)

Here, we propose a method that can recognize an attack as an incorrect evaluation that does not exceed the threshold. In Proposed Method (1), a collusion attack in which the evaluation values do not differ by more than a standard deviation from the previous values may not be recognized. In order to resist such an attack, we propose a scheme that assigns each evaluation a weight that is based on its difference from the community trend; that is, the weight is based on the deviation of e_{ij} from $Ave[j]$. If the deviation exceeds $Th[i]$, the weight becomes 0. This way of assigning weights is shown in **Fig. 7**. Note that the influence of an attack that uses a value near $Th[i]$ is suppressed. The algorithm that includes this assignment of weights is shown in **Fig. 8**.

In Fig. 8, E'' is the new evaluation matrix, with elements e''_{ij}, which are calculated using the weight $(1 - |e_{ij} - Ave[j]|/Th[i])$. If $|e_{ij} - Ave[j]| > Th[i]$, then e''_{ij} becomes $Ave[j]$. Therefore, the value of any evaluation that is near $Th[i]$ is reduced. In this way, this method is able to defend from an attack that uses an evaluation value that is less than $Th[i]$.

4. Simulation

4.1 Simulation Conditions

In order to evaluate the proposed method, we conducted simulations with the following conditions:

(1) Synthesized dataset:

The numbers of agents and objects were each 100. The numbers of links in the provisioning matrix and the evaluation matrix were each 10,000. Normal agents provided evaluations that were equal to the average value plus a random factor of up to 20%. Malicious agents assigned to Object 1 an evaluation that is five times higher than the true evaluation value. The percent of the agents who were malicious was varied from 0% to 50%. The standard deviation of the evaluations was 1.0.

(2) Public reputation dataset:

For this simulation, we used the MovieLens [15] dataset, which is a collection of evaluations of movies. There are 100,000 objects, 943 agents, and 1,682 movies. Since this dataset does not have the same structure as our model, we could not apply our algorithm directly. This dataset contains only evaluation values, and so the provisioning matrix is not defined. Therefore, we created the provisioning matrix from the evaluations; that is, from the evaluations, we determined the information provided to the user, and thus created the matrix P. We then assumed that malicious agents give to the first movie a rating that was five times higher than the true evaluation. The percent of the agents who were malicious was varied from 0% to 50%. The standard deviation of the evaluations was 1.0.

4.2 Simulation Results

(1) Synthesized dataset:

In order to select the best standard deviation, we ran simulations with EigenRumor and Proposed Method (1) in which the evaluations ranged from 0 to 10. 10% of the agents were malicious. **Figure 9** shows the results of the simulation, and **Table 1** shows the rate at which the malicious evaluations were detected, and reports the numbers of true positives (TPs; correctly identified as malicious), false positive (FPs; incorrectly identified as malicious), false negatives (FNs; incorrectly identified as not malicious), and true negatives (TNs; correctly identified as not malicious). Since 10% of the agents were malicious and only one node was targeted, there were ten malicious evaluations.

From Table 1, we see that $\tau = 2.0$ is the best. However, from Fig. 9, we see that $\tau = 1.0$ gives the best reputation value, so we will use this value.

Figures 10 and **11** present the reputation results of the simulations for all objects using Proposed Method (1) and EigenRumor. In the results shown in Fig. 10, no agents were malicious, and the results are the same for both methods; that is, Proposed Method (1) behaves correctly when there are no malicious agents. In the

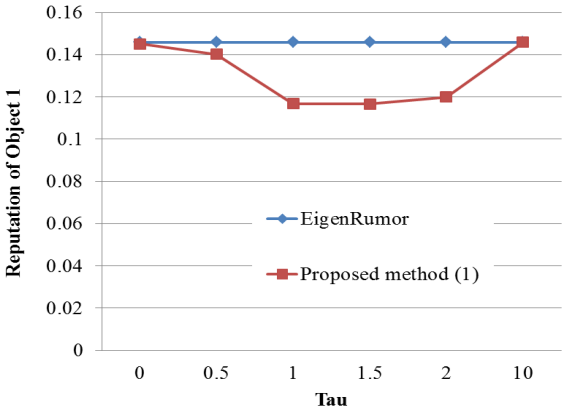


Fig. 9 Simulation results of Proposed Method (1) and EigenRumor for τ.

Table 1 Detection rate of malicious evaluations (TP = true positive; TN = true negative; FP = false positive; FN = false negative).

τ	TP	FP	FN	TN
0.0	10	9990	0	0
0.5	10	8785	0	1205
1.0	10	4008	0	5982
1.5	10	1276	0	8714
2.0	10	75	0	9915
10.0	0	0	10	9990

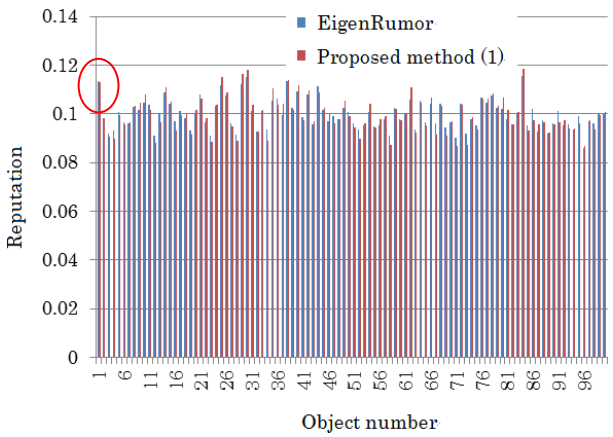


Fig. 10 Simulation results of Proposed Method (1) and EigenRumor (no malicious agents).

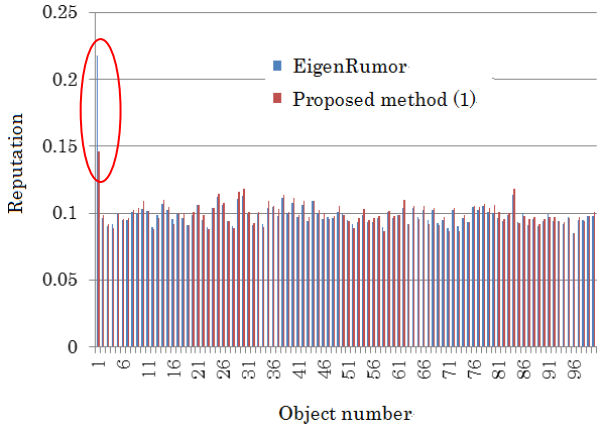


Fig. 11 Simulation results of Proposed Method (1) and EigenRumor (30% of the agents are malicious).

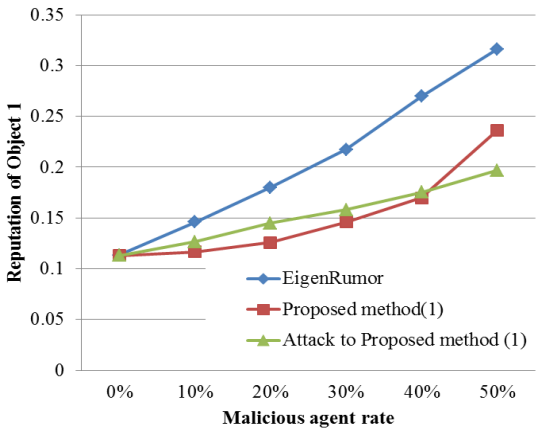


Fig. 12 Simulation results of Proposed Method (1).

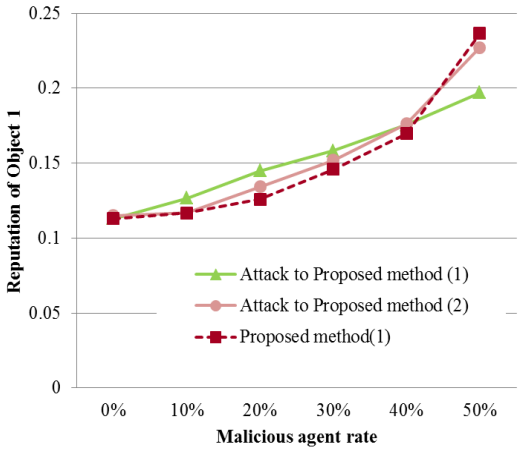


Fig. 13 Simulation results of Proposed Method (2).

results shown in Fig. 11, 30% of the agents were malicious, and the results of the two methods are the same except for Object 1. For that object, the reputation value obtained by EigenRumor is about 120% larger than that given to the other objects. However, with Proposed Method (1), it is only about 45% larger. Therefore, the effect of the attack was reduced.

Figure 12 shows the reputation values assigned to Object 1 in this simulation. The horizontal axis shows the percentage of agents that were malicious. The reputation assigned by Proposed Method (1) is lower than that assigned by EigenRumor, even if the percentage of malicious agents increases. Following an attack, Proposed Method (1) lowers the reputation value.

Figure 13 shows the reputation values assigned by the two proposed methods to Object 1 when there was an attack. When 0% to 40% of the agents were malicious, the value assigned by Proposed Method (2) to the attack victim was lower, but it was higher than that assigned by Proposed Method (1). When 50% of the agents were malicious, Proposed Method (2) gave a lower value than did Proposed Method (1). However, since 50% is half of the community, a malicious attack has no meaning. Therefore, we conclude that Proposed Method (2) performs better.

(2) Public reputation dataset:

Table 2 shows the simulation results for the public reputation dataset. In general, as the percentage of malicious agents increases, the reputation and the rank of the target both increase. However, the proposed method is able to maintain the reputation

Table 2 Reputation value and rank of the target item (Simulation results with the public reputation dataset).

Malicious rate	EigenRumor		Proposed	
	Reputation	Rank	Reputation	Rank
0%	1.14E-04	1041	1.13E-04	1046
10%	1.58E-04	972	1.21E-04	1040
20%	2.19E-04	859	1.11E-04	1055
30%	2.61E-03	75	1.11E-04	1055
40%	8.37E-01	1	1.67E-02	1
50%	9.50E-01	1	2.05E-02	1

at almost the same value as the percentage of malicious agents grows from 0% to 30%. From this simulation, we see that the proposed method is applicable to this dataset.

5. Conclusions

We proposed an algorithm that will improve the collusive attack resistance of EigenRumor. In this paper, we show that EigenRumor is resistant to a Sybil attack, but not to a collusive attack on a specific target. In this paper, we considered the case in which malicious agents collude to target the reputation of a single object. If an agent provides an evaluation that is very different from the community trend, it is discarded; that is, malicious agents are not able to provide evaluations that differ greatly from those given by the rest of the community. The threshold for invalidating an evaluation is based on the previous differences between evaluations in that community.

To increase this threshold, there must be more than one object that receives an unusually high evaluation. However, this will cause EigenRumor to decrease the hub score (ability) of that agent, and so the attack to raise the threshold will fail.

In order to evaluate the effectiveness of the proposed method, we conducted simulations, and the results show that the proposed method performs better than the original EigenRumor algorithm. Furthermore, Proposed Method (2) is resistant to an attack to which Proposed Method (1) is vulnerable. The simulation results of the public reputation dataset show that our algorithm is effective and applicable to data that have a different structure than that assumed in the original EigenRumor model.

Our intended future work is to develop a more precise algorithm that is based on the time sequence of the evaluations. In order to apply it to the Wireless Sensor Networks (WSN) and ad hoc networks, it will be necessary to develop a robust reputation system.

Acknowledgments Part of this work was carried out under the Cooperative Research Project Program of the Research Institute of Electrical Communication, Tohoku University.

References

[1] Swamynathan, G., Almeroth, K.C. and Zhao, B.Y.: The design of a reliable reputation system, *Electronic Commerce Research*, Vol.10, No.3-4 (2010).

[2] Jøsang, A. and Golbeck, J.: Challenges for Robust Trust and Reputation systems, *5th International Workshop in Security and Trust Management* (*STM2009*) (2009).

[3] Jøsang, A., Ismail, R. and Boyd, C.: A survey of trust and reputation systems for online service provision, *Decision Support Systems*, Vol.43, No.2 (2007).

[4] Suryanarayana, G. and Taylor, R.N.: A Survey of Trust Management and Resource Discovery Technologies in Peer-to-Peer Applications, ISR Technical Report #UCI-ISR-04-6, University of Calfornia, Irvine (2004).

[5] Ushikubo, H., Takeda, S. and Shigeno, H.: A Reputation Aggregation Method Considering File Request Distribution and Peer Behavior in Unstructured P2P Networks, *7th International Conference on Broadband, Wireless Computing, Communication and Applications* (*NBiS2012*), pp.74–81 (2012).

[6] Selvaraj, C. and Anand, S.: A Survey on Security Issues of Reputation Management Systems for Peer-to-Peer Networks, *Computer Science Review*, Vol.6, No.4, pp.145–160 (2012).

[7] Buchegger, S. and Le Boudec, J.-Y.: Performance analysis of the CONFIDANT protocol, *Proc. 3rd ACM International Symposium on Mobile Ad hoc Networking & Computing*, pp.226–236 (2002).

[8] Mundinger, J. and Le Boudec, J.-Y.: Analysis of a reputation system for Mobile Ad-Hoc Networks with liars, *Performance Evaluation*, Vol.65, No.3-4 (2008).

[9] Khalid, O., Khan, S.U., Madani, S.A., Hayat, K., Khan, M.I., Min-Allah, N., Kolodziej, J., Wang, L., Zeadally, S. and Chen, D.: Comparative study of trust and reputation systems for wireless sensor networks, *Security and Communication Networks*, Vol.6, No.6, pp.669–688 (2013).

[10] Michiardi, P. and Molva, R.: Core: A collaborative reputation mechanism to enforce node cooperation in mobile ad hoc networks, *Proc. IFIP TC6/TC11 Sixth Joint Working Conference on Communications and Multimedia Security: Advanced Communications and Multimedia Security*, pp.107–121 (2002).

[11] West, A.G., Aviv, A.J., Chang, J. and Lee, I.: Spam Mitigation Using Spatio-temporal Reputations from Blacklist History, *Proc. 26th Annual Computer Security Applications Conference*, *ACSAC'10*, pp.161–170 (2010).

[12] Fujimura, K. and Tanimoto, N.: The EigenRumor Algorithm for Calculating Contributions in Cyberspace Communities, *Lecture Notes in Computer Science*, Vol.3577, pp.59–74 (2005).

[13] Fujimura, K., Inoue, T. and Sugisaki, M.: The EigenRumor Algorithm for Ranking Blogs, *WWW Workshop on the Weblogging Ecosystem* (2005).

[14] Brin, S. and Page, L.: The Anatomy of a Large-scale Hypertextual Web Search Engine, *Proc. 7th International World Wide Web Conference* (1998).

[15] MovieLens 100k, available from ⟨http://grouplens.org/datasets/movielens/⟩.

Fumiaki Sato received his B.E. degree from Iwate University, Japan, in 1984. He received his M.E. and D.E. degrees from Tohoku University, Japan, in 1986 and 1992 respectively. In 1986, he joined Mitsubishi Electric Corporation. From 1995–2005, he worked Shizuoka University. He is now a Professor in the department of Information Science of Toho University, Japan. He teaches undergraduate and graduate in Computer Networks, Mobile Computing, and Ad-hoc Networks. His research areas of interest include communication protocols, ad-hoc networks, sensor networks, P2P systems, network security, distributed processing systems, and communication software design. He is a member of IPSJ, IEICE, IEEE and ACM.