 CMC, vol.58, no.3, pp.845-857, 2019

Dynamic Trust Model Based on Service Recommendation in Big Data

Gang Wang[1, *] and Mengjuan Liu[2]

Abstract: In big data of business service or transaction, it is impossible to provide entire information to both of services from cyber system, so some service providers made use of maliciously services to get more interests. Trust management is an effective solution to deal with these malicious actions. This paper gave a trust computing model based on service-recommendation in big data. This model takes into account difference of recommendation trust between familiar node and stranger node. Thus, to ensure accuracy of recommending trust computing, paper proposed a fine-granularity similarity computing method based on the similarity of service concept domain ontology. This model is more accurate in computing trust value of cyber service nodes and prevents better cheating and attacking of malicious service nodes. Experiment results illustrated our model is effective.

Keywords: Trust model, recommendation trust, content similarity, ontology, big data.

1 Introduction

Trust-based security system has been an important research field nowadays since trust management was proposed by Blaze et al. [Blaze, Feigenbaum and Lacy (1996)] in 1996. When big data went into our life, work environment and physical world, trust management about big data service became an important issue [David and Ongand (2016); Liu, Dong, Ota et al. (2016); Siddiqa, Hashem, Yaqoob et al. (2016)]. Trust management involves a lot of fields, and some scholars have already been going studies in these issues and proposed a series of trust models in accordance with different application systems. Zhang et al. observed that "trust model was separated into policy-based and reputation-based models in terms of management mechanism in e-commerce environment" [Zhang, Cheng, Jiang et al. (2008); Nejdl, Olmedilla and Winslett (2004); Chu, Feigenbaum, LaMacchia et al. (1997)]. Trust model was separated into two trust models based on centralization and distribution in a view of structure [Yu and Singh (2002); Resnick and Zeckhauser (2002); Zacharia, Moukas and Maes (2000)]. Li et al. proposed "*PeerTrust* Model: an electronic community based on local reputation under P2P environment" [Li and Liu (2004)]. Dou [Dou, Wang, Jia et al. (2004)] proposed "a global trust model that is intended to overcome the lack of security of literature" [Zhang,

[1] Xi'an University of Finance and Economics, Xi'an, 710100, China.

[2] Shaanxi Radio and Television University, Xi'an, 710068, China.

[*] Corresponding Author: Gang Wang. Email: wgang_only@126.com.

Cheng, Jiang et al. (2008)]. Literature [Li, Jing, Xiao et al. (2007); Jin, Zhang, Qu et al. (2008)] investigated a similarity-based added-weight trust model that can help solving the problem of malicious nodes cheating.

From these literatures' reviews, we found that they all lacked to consider difference among different services. In addition, we also found that existing trust models lacked fine granularity computing, so it is difficult to distinguish service difference among different service providers, and difference among different services of the same service provider. We proposed a dynamic trust model based on service recommendation in big data. The other parts of this paper are organized as follows: Section 2 is dynamic Trust model. Section 3 is concept similarity computing of trade goods (or services) domain ontology and trust model algorithm. Section 4 is simulation experiment and result analysis. Final section is next works and conclusions.

2 Dynamic trust model

For clearly illustrating own model, we firstly gave a few related concepts as follows.

Definition 1. Service requestor (service receiver) is to get services from network service provider.

Definition 2. Service provider is to provide services for network service receivers.

Definition 3. Node that recommends information to Service requestor is called recommendation node, and then they are classified to direct familiar recommendation node, indirect familiar recommendation node and stranger recommendation node.

Definition 4. Direct familiar recommendation node refers to have ever direct services between recommendation node and service provider.

Definition 5. Indirect familiar recommendation node refers to have already had direct transactions or indirect transactions with direct familiar recommendation nodes.

Definition 6. Direct Trust is that service requestor gives an evaluation of service provider and this evaluation is based on direct services that service provider gave service requestor ago.

Definition 7. Recommendation trust refers to an evaluation of service requestor to service provider according to their history service state.

2.1 Trust computing model

System would like to judge the whole trust value of node k (node k is a service provider or resource provider) is closely related to two factors. One factor is that whole trust value of node k usually comes from evaluation of other nodes to it, and whether this evaluation is accuracy and objective is largely depended on similarity between services or service received by individual evaluating node. The other is that whole trust value of node k is also related to a trust evaluator's familiarity with a trust recommendation node (also known as resource recommendation node). The structure of this model is described in Fig. 1.

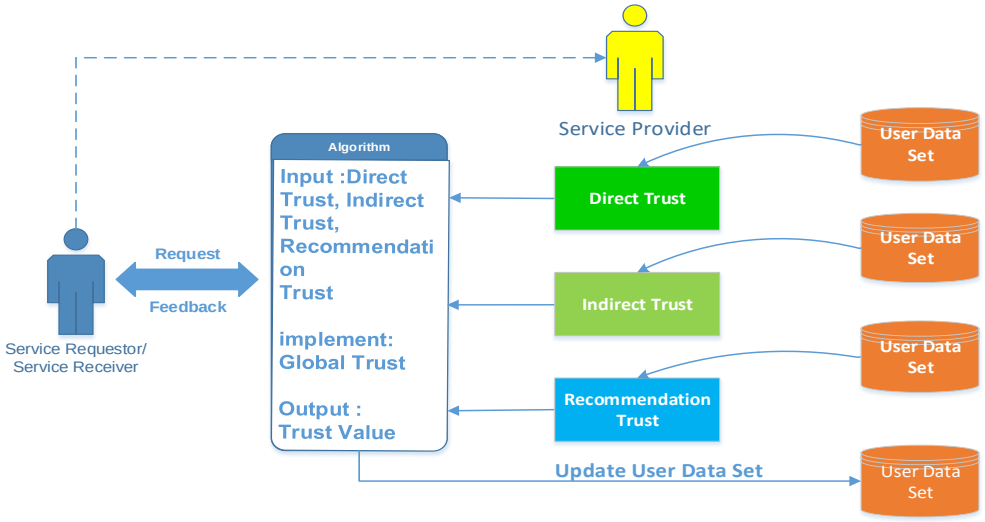

Figure 1: Trust computing model

From Fig. 1, when Service Requestor applied a service to Service Provider, system computed and fed back trust value of Service Provider, and then gave an advice whether Service Requestor selected service proposed by Service Provider. During computing trust, algorithm will firstly fuse Direct Trust, Indirect Trust and Recommendation Trust, in the process of fusion we will also comprehensive consider relationship among Direct Trust, Indirect Trust and Recommendation Trust because trust in itself involves multi-attribution such as service content similarity, time sensitive. Meanwhile, system updated trust value of Service Provider in User Dataset.

2.2 Direct trust computing

When there was history interaction between service provider and service receiver, we use following equation as Direct Trust.

$$DT_{i,j} = \frac{S_{i,j}}{G_{i,j}} = \frac{S_{i,j}}{S_{i,j} + F_{i,j}} \tag{1}$$

In which $DT_{i,j}$ is direct trust value, $S_{i,j}$ is the successful number of times between i and j. $G_{i,j}$ is all interactions times between i and j. $F_{i,j}$ is the failures number of times between i and j. when $G_{i,j} = 0$, we set $DT_{i,j} = 0.5$. Meanwhile, $DT_{i,j} \neq DT_{j,i}$.

2.3 Recommendation trust computing

Recommendation Trust computing equation that we used is as following:

$$\overline{RT}_k = \sum_{\substack{i,j=1 \\ i \neq j}}^{n} Sim(C_{i,k}, C_{j,k}) * (\alpha * \omega_{i,k} * DT_{i,k} + \beta * \omega_{j,k} * DT_{j,k}) / n \tag{2}$$

where $\overline{RT}_k$ refers to integration evaluation value of recommendation trust to node k, and $Sim(C_{i,k}, C_{j,k})$ is the similarity of transaction content $C_{i,k}$ and $C_{j,k}$ of between node *i*, *j* and *k*, *j* and *k*, ω_{ik} and ω_{jk} respectively refer to acquaintance recommendation weight and stranger recommendation weight, and ω_{ik} refers to evaluation value of service requestor to service recommendation node; in which ω_{ik} and ω_{jk} , $\begin{cases} \omega_{ik} = DT_{i,k} \\ \omega_{jk} = 0.5 \end{cases}$. The recommendation weight to stranger is set at 0.5, that is to say, trust and un-trust of node initially are fifty-fifty. α is the recommendation weight of direct or indirect familiar node, and then α is set at as following.

$$\begin{cases} \alpha = DT_{i,j} \\ \alpha = \frac{1}{n} \sum_{N=1, i \neq j \neq k}^{n} \prod DT_{i,j} * DT_{i,j} = DT_{i,j} \times DT_{j,k} \times DT_{k,l} \times \cdots \end{cases} .$$

If α<ε, this node is abandoned. ε refers to threshold value of trust chain and was set by us. β is recommendation weight of stranger. If $\beta = \frac{\sum_{i=1}^{n} DT_{ij}}{n}$, while $\sum_{i=1}^{n} DT_{ij} = 0$, this node is actually a new joining node or a dormant node.

$\alpha * \omega_{i,k} * DT_{i,k}$ and $\beta * \omega_{j,k} * DT_{j,k}$

compute trust degree of node whose three square-root is used to ensure the weight of each coefficient can be within a normal range. At the same time, it also ensures that system can be easy to make reasonable trust judgment to node. In this case, Eq. (2) becomes Eq. (3).

$$\overline{RT_{i,k}} = \frac{\sum_{\substack{i,j=1 \\ i \neq j}}^{n} Sim\left(C_{ik}, C_{jk}\right) \times \left(\sqrt[3]{\alpha \times \omega_{ik} \times DT_{i,k}} + \sqrt[3]{\beta \times \omega_{jk} \times DT_{j,k}}\right)}{n} \tag{3}$$

2.4 Global synthetical trust

Global synthetical trust refers to a general trust degree. The paper used following equation to compute Global Trust Degree.

$$GT = \begin{cases} RT, & if\ s = 0 \\ DT, & if\ \tau = 0 \\ \rho * DT + (1 - \rho) * RT, & if\ s \neq 0\ and\ \tau = 0 \end{cases} \tag{4}$$

In which, *s* is the number of recommendation node, τ is the number of direct transaction between trust evaluator and service provider. ρ is the weight of direction transaction, $1 - \rho$ is the weight of recommendation trust. With social relationship analysis, trust earned by direct interaction of two nodes is higher than trust earned by nodes recommendation, so trust evaluator believes target node by interaction of self-trust evaluator and target node. $\rho(x)$ is a function to indicate that trust is a dynamic change with interactive numbers and we make argument in Eq. (5).

$$\rho(x) = 1 - \left(\frac{1}{2}\right)^{\frac{x}{n-x}},\ (n-x) \neq 0 \tag{5}$$

$n \in \{1,2,3, \ldots\}$, x refers to the *x-th* interaction between service evaluator and service provider. x and ρ are positive correlation. When service requestor is the first interaction with service provider, because the number of interactions is too little, service requestor has to rely on recommendation trust evaluation given by the others. When the number of interactions is creasing between service requestor and service provider, service requestor would like to judge trust value of service provider by its own, and then ρ will enlarge gradually as increase of interactive numbers.

3 Computing of Concept similarity degrees about service domain ontology

3.1 Service or trade goods ontology building process

This paper proposed a new service domain ontology building method. We firstly gave its mainly building process as follows:

3.1.1 Aim of building domain ontology

Building different domain ontology is for different application purpose in different field. Thus, the first step of building domain ontology to our model aims to complete our application purpose.

3.1.2 Determining ontology Concept and classification system

There generally are two ways to get domain ontology data sources. "The first comes from lexicons and professional dictionaries. The second relies on mathematical analysis by using set of language data and documents, and computing and analyzing the weight of conceptualizations, and then selecting concepts of those greater weight" [Jia (2006); Uschold (1996)]. This paper uses the principles of Chinese Library Catalogues, Trade Marks and Commodity Catalogues of the People's Republic of China as building way of service domain ontology, and classification and definition of Alibaba[2], Taobao[3] and Baidu on E-commerce products. Determining attribution and relationship among classes.

Class attribution is as a classifying basis of research object, and is used to distinguish between different classes, and a subclass will inherit attributions of father class. Besides defined class attributions, it also added constraints to specific attribute during building ontology. Constraints may be derived from father class, and there are also some new attribution constraints to a subclass. e.g., the key attributes of automobile ontology is those attributes such as design and production. However, the key attributes of automobile as commodities focus more on its applications and the attributes as commodity, such as price, color and exterior. In addition, Relationship among classes is also an attribution. Fig. 2 is a service domain ontology structure built by us.

[2] http://china.alibaba.com/

[3] http://www.taobao.com/

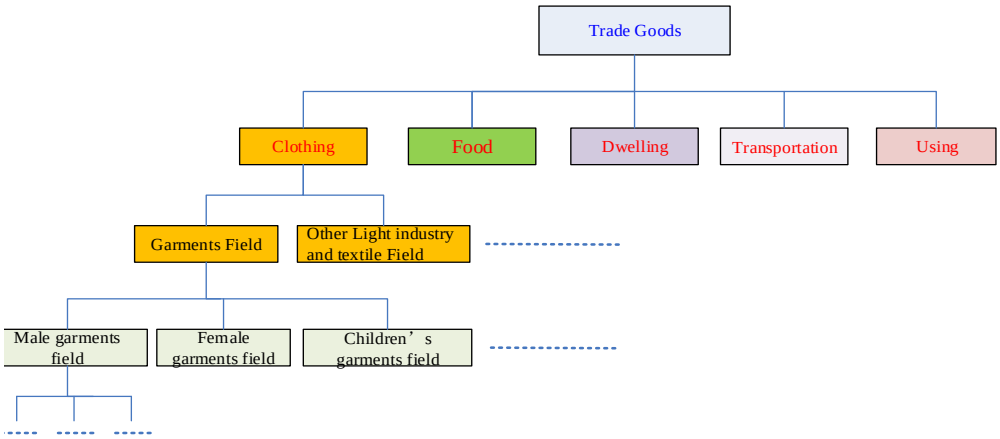

Figure 2: Service domain ontology structure

3.1.3 Formal definition and explanation of ontology

Definition 8: ontology indication uses five tuples,

$$O_{product}=\left\{C_{product},R_r,A^{C_{product}},A^{R_r}_{product},X_{product}\right\} \tag{6}$$

Of which, $C_{product}$ is concept set; R_r is a set of relationship; $A^{C_{product}}$ is a concept attribution set composed by more attribution sets; $A^{R_r}_{product}$ refers to a set of more attributions component and is also a relationship set of concepts; $X_{product}$ refers to an axiom set.

Definition 9: if $C^i_{product}$ and $C^j_{product}$ belong to concept $C_{product}$,and denoted by $C^i_{product},C^j_{product}\in C_{product}$, values of relationship R_r arranges from $C^i_{product}$ to $C^j_{product}$, that is $r=\left\langle C^i_{product},C^j_{product}\right\rangle$, concept $C^i_{product}$ and $C^j_{product}$ have relationship R_r, and denoted by $C^i_{product}R_rC^j_{product}$, and $C^j_{product}\in\left\{C^i_{product}\right\}*r$, $C^i_{product}\in r*\left\{C^j_{product}\right\}$. In which, indicates $C_{product}\times R\to C_{product}$, * indicates $R\times C_{product}\to C_{product}$, $C_{product}$ presents a concept domain, R presents relationship domain. Inverse relationship of R_r presented:

$R_r^{-1},r^{-1}=\left\langle C^j_{product},C^i_{product}\right\rangle$, and then

$C^i_{product}\in\left\{C^j_{product}*r^{-1}\right\},C^j_{product}\in\left\{r^{-1}*C^i_{product}\right\}.$

Definition 10: if concept $C^i_{product}$ belong to concept $C_{product}$, and is denoted $C^i_{product}\in C_{product}\ (i\in N)$, $A^{Cproduct}$ presents a set to be composed of attributions, where there is a match between each of attributions and a concept.

$$A^{(C_{product})} = A^{(C_{product})}(C^1_{product}), A^{(C_{product})}(C^2_{product}), \cdots, A^{(C_{product})}(C^i_{product}) \quad (7)$$

Definition 11: if concept $C^i_{product}$ and $C^j_{product}$ belong to concept $C_{product}$ and is denoted by $C^i_{product}$, $C^j_{product} \in C_{product}$, and $A^{(C_{product})}(C^i_{product}) \subseteq A^{(C_{product})}(C^j_{product})$, $C^j_{product} \in is_{subclass_of} * C^I_{product}$ or $\langle C^j_{product}, C^i_{product} \rangle \in \langle C^i_{product}, C^j_{product} \rangle$ *is_subclass_of*.

Definition *12: if* $C^i_{product}, C^j_{product} \in C_{product}$ *to concepts* $C^i_{product}$ *and* $C^j_{product}$, *and* $A^{C_{product}}(C^i_{product}) \supseteq A^{C_{product}}, C^i_{product} \equiv C^j_{product}$.

Definition *13: if* $C^i_{product} \subseteq \neg C^j_{product}$ *to concepts* $C^j_{product} \subseteq \neg C^i_{product}$, $(C^i_{product}, C^j_{product}) \in Disjoint\ with$.

3.1.4 Ontology coding

Trade goods ontology adopts Protégé which is a software development tool, and it can create ontology and run created ontology. In which, Fig. 3 is a part of trade goods ontology.

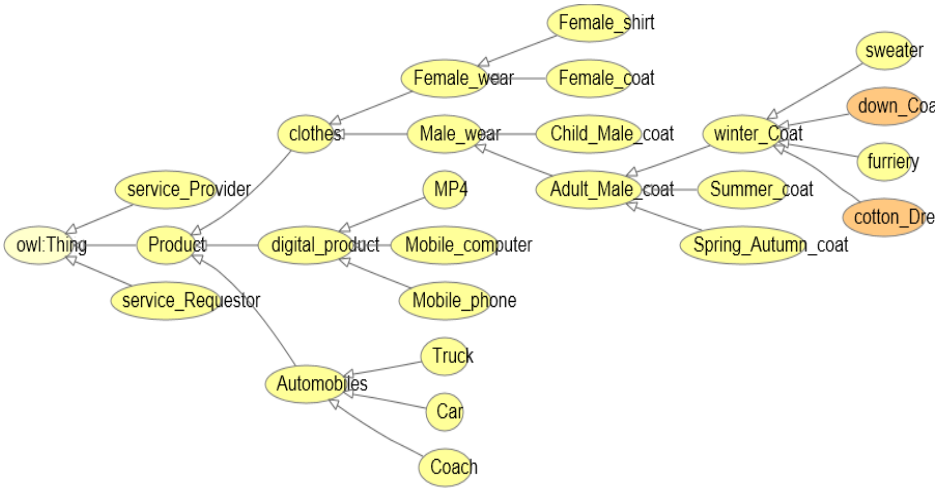

Figure 3: Part of trade goods ontology picture

3.1.5 Ontology assessment

Ontology assessment is focused on clarity, coherence and extendibility. Moreover, other issues would be thought such as encoding minimal mistake probability, minimum restrict of ontology, etc. [Gruber (1995); Wu, Wu, Li et al. (2005)].

3.2 Similarity computation of ontology Concept in service domain

This paper put forward a similarity computation way of ontology concept. This method used statistical information entropy to calculate the weight of evaluation index and

avoided the subjectivity of artificial weight method. At the same time, this paper also used heuristic algorithm to ensure a clear difference between two or more similar ontology concepts.

In ontology concept, concept C_i has to satisfy Eq. (8):

$$I(C_i) = I(C_i^{superclass}) + I(C_i^{self} \mid C_i^{superclass}) \tag{8}$$

Of which, $I(C_i^{superclass})$ denotes the information of $C_i^{superclass}$ that is parent class of C_i; $I(C_i^{self} \mid C_i^{superclass})$ denotes self-information of C_i.

$$I\left(C_i^{self} \mid C_i^{superclass}\right) = I(C_i) - I\left(C_i^{superclass}\right) \tag{9}$$

$$p\left(C_i^{self} \mid C_i^{superclass}\right) = \frac{p(C_i)}{p(C_i^{superclass})} \tag{10}$$

$p(C_i)$ is that concept C_i appears probability in ontology concept, because sub-concept inherits some information from parental concept, there is the following Eq. (11).

$$p(C_i) = \frac{\sum n(C_i^{subclass}) + 1}{n(o)} \tag{11}$$

$\sum n(C_i^{subclass})$ refers to sub-concept numbers of concept C_i, $n(o)$ indicates concept numbers of ontology *O*. Information contained by two concepts decides semantic distance between them, hence there is Eq. (12):

$$dis(C_i, C_j) = \sum_{this=i}^{d_{root}} f(C_{this}, C_{this}^{superclass}) \times (I(C_{this}) - I(C_{this}^{superclass})) + \sum_{this=j}^{d_{root}} f(C_{this}, C_{this}^{superclass}) \times (I(C_{this}) - I(C_{this}^{superclass})) \tag{12}$$

$f(c_{this}, c_{this}^{superclass})$ presents the weight function along $C_{this} \to C_{this}^{superclass}$, where C_{this} is a node in this path, $C_{this}^{superclass}$ is a parental node of C_{this}, computation equations is Eq. (13).

$$f(C_{this}, C_{this}^{superclass}) = \frac{I(C_{this}^{superclass}) - I(C_{this})}{I(C_{i,j}^{droot}) - I(C_{\kappa})} \tag{13}$$

$C_{i,j}^{droot}$ is a parental node of concepts C_i and C_j, difference between two concepts can bring about change of $C_{i,j}^{droot}$ on the shortest path between them. C_{κ} presents a node of same side of C_{this}. Two concepts similarity computing formula is as follows:

$$Sim(C_i, C_j) = \frac{1}{1 + dis(C_i, C_j)} \tag{14}$$

As is well known, when information of two concepts is respectively same, we think two concepts are the same, for avoiding the situation, we used heuristic algorithm to compute similarity. Heuristic algorithm is Tab. 1.

Table 1: Heuristic Algorithm

Heuristic Algorithm	Conditions
HA1	*HasSameSuperclass()*
HA2	*HasSameSiblingclass()*
HA3	*HasSameSubclass()*
HA4	*HasSameInstance()*
HA5	$\forall c_i^{subclass}$ *similarity X,* $\exists c_i^{superclass}$ *similarity X*
HA6	$\forall c_i^{sibling}$ *similarity Y,* $\exists c_i$ *similarity Y*

3.3 Trust computing algorithm

Initialization: To set the total number of node and service
Target: To get trust degree of service provider

Step 1: To initialize service path map. To define all network nodes. In which, there are never any interaction before between any two nodes. Meanwhile, to set the number of malicious nodes and good nodes. And then services are randomly distributed to nodes. When a node is defined malicious node, its service only has a half of a good node. Finally, system will create a service path map.

Step2: To compute Global synthetical trust degree, we need firstly initialize ε, i, j. Secondly, while i is larger than the numbers of loop and computing trust degree is larger than ε, system adopts this service path map, unless system cancels this service path map.

Step3: If i is smaller than the total number of service path and there aren't malicious nodes in the present service path, system uses *successNum* as successful service number and uses *totalNum* as service total number.

Step4: If j is smaller than the number of service path and i and j are two different nodes, i and j are going to mutual service, and service successful ratio is the number of successful services dived by the total number of services, unless system abandons node j and algorithm go back to Step 2.

4 Simulation experiment and result analysis

We divide service nodes into two sorts of nodes, one is good nodes, and the other is malicious nodes. Meanwhile, malicious nodes are divided into individual malicious nodes and cooperative malicious nodes.

4.1 Experiment analysis

Experiment 1: Analysis to malicious recommendation nodes scale.

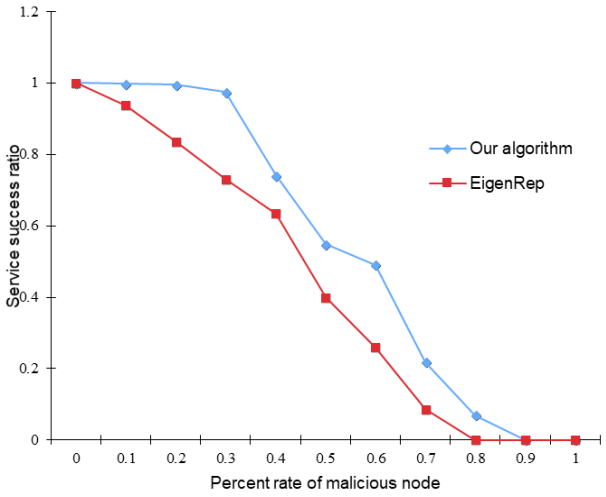

Figure 4: Service success ratio in malicious nodes dynamic change

Results of the experiment from Fig. 4 show whole tendency to two algorithms is similar, e.g., with increasing of percent rate of malicious node, we find service ratio of two algorithms declines gradually. However, there are a few important differences between two algorithms. Firstly, when malicious node rate is from 0 to 30%, we find that our algorithm effect is better than EigenRep algorithm. Service success ratio of our algorithm seldom decline, but EigenRep success ratio declines rapidly. From 30% malicious nodes to 70% malicious nodes, we can find though success ratio of two algorithms all declines, our algorithm effect is better than EigenRep algorithm. Secondly, when we set percent rate of malicious nodes is 50%, we find our algorithm still has a high service success rate. In a word, from these falling service success rate and numbers, no matter what it is an individual malicious node or cooperative malicious node, we can find our algorithm is better than EigenRep algorithm.

Experiment 2: Sensitivity analysis to number of service cycle.

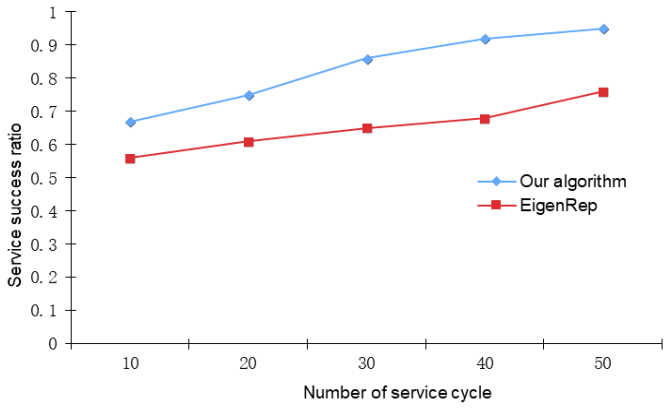

Figure 5: Service success rate with different service circulation times

In this experiment, malicious node rate we firstly set is 40%. We can find tendency of two algorithms is increase gradually with increasing of service circulation times, that is to say, tendency of two algorithms is similar. However, there are still a few differences because we can find service success rate of two algorithms is different in the different circulation times. Firstly, in 50-th cycle, we find EigenRep algorithm's service success rate is only 69.4%, but our algorithm is 90.2%, our algorithm effect is obviously better than EigenRep. Secondly, our algorithm can constrain better cheat and pretending of malicious node. Obviously, from Fig. 5 we can find that sensitivity of our algorithm to malicious attack is better than EigenRep's.

Experiment 3: Analysis directed to malicious recommendation attack.

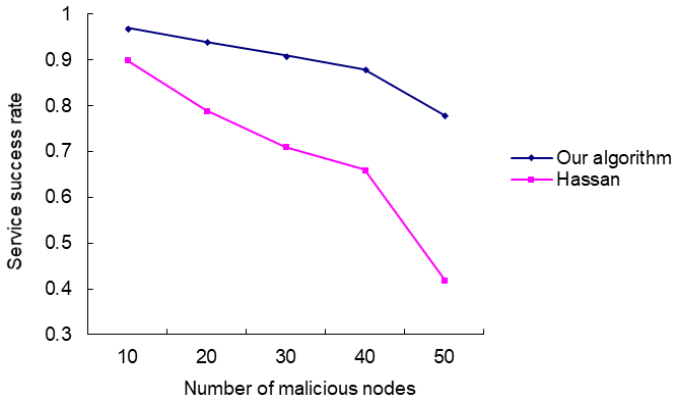

Figure 6: Service success rate in different malicious recommendation nodes

This picture illustrates that tendency of two algorithms is decline gradually with the increasing number of malicious recommendation nodes, and we find that tendency of two algorithms is similar. However, there are still a few differences because we can find service success rate of two algorithms is different under different malicious node numbers. When the number of malicious nodes is 30%-40%, we find Hassan algorithm's service success rate is obviously lower than our algorithm. This experiment illustrates our algorithm effect is obviously better than Hassan. Secondly, though Hassan model is good at constraint to node cheat, Hassan model has a few lacks, e.g., Hassan has an assumption that recommendation nodes is trust, and Hassam does not have a specific punishment strategy, so its effect is not good enough to constrain malicious recommendation.

By these analyses, we can know trust algorithm proposed in this paper can effectively constrain malicious recommendation of dishonest nodes, and in same recommendation road good nodes can ensure service success rate. By compare our model with EigenRep and Hassan algorithms, experiment results illustrate our model is good enough to constrain malicious recommendation, cheat and attack.

5 Conclusions and further works

In big data environment trust is one of all preconditions in network service. Under the circumstances, we proposed trust evaluation model based on service recommendation. This model could distinguish familiar nodes from stranger nodes in cyber, and also took

into account service or transaction ontology in business to make sure computing accuracy of recommendation trust, and computing effect was also better than past some trust models. Experiment results illustrated this algorithm effect is better than several typical trust models like EigenRep and Hassan model. At the same time, we'll consider to improve algorithm efficiency and introduce more characteristics of trust so that trust compute is more accurate in further works.

Acknowledgment: This paper Supported by Natural Science Basic Research Plan in Shaanxi Province of China (Program No. 2014JM2-6099) and the Project of Xi'an University of Finance and Economics (No. 17FCJH13).

References

Blaze, M.; Feigenbaum, J.; Lacy, J. (1996): Decentralized trust management. *Proceedings 1996 IEEE Symposium on Security and Privacy*, pp. 164-173.

Chu, Y.; Feigenbaum, J.; LaMacchia, B.; Resnick, P.; Strauss, M. (1997): Referee: trust management for web applications. *World Wide Web Journal*, vol. 2, no. 2, pp. 127-139.

David, G.; Ongand, Y. S. (2016): Modeling and management of big data: challenges and opportunities. *Future Generation Computer Systems*, vol. 63, no. 10, pp. 96-99.

Dou, W.; Wang, H. M.; Jia, Y.; Zou, P. (2004): A recommendation-based peer-to-peer trust model. *Journal of Software*, vol. 15, no. 4, pp. 571-583.

Gruber, T. R. (1995): Towards principles for the design of ontologies used for knowledge sharing. *International Journal of Human-Computer Studies*, no. 43, pp. 907-928.

Jia, L. L. (2006): *Research of Relation Between Concepts in Ontology Building (Ph.D. Thesis)*. Graduate School of Chinese Academy of Agricultural Sciences, Beijing.

Jin, Y. W.; Zhang, Y.; Qu, W. Y.; Liu, Y; Li, K. (2008): A trust model based on similarity evaluation in P2P networks. *Proceedings of the 2008 IEEE International Symposium on Parallel and Distributed Processing with Applications*, pp. 737-742.

Li, J. T.; Jing, Y. N.; Xiao, X. C.; Wang, X. P.; Zhang, G. D. (2007): A trust model based on similarity-weighted recommendation for P2P environments. *Journal of Software*, vol. 18, no. 1, pp. 157-167.

Liu, X.; Dong, M. X.; Ota, K.; Hung, P. ; Liu, A. F. (2016): Service pricing decision in cyber-physical systems: insights from game theory. *IEEE Transactions on Services Computing*, vol. 9, no. 2, pp. 186-198.

Nejdl, W.; Olmedilla, D.; Winslett, M. (2004): PeerTrust: automated trust negotiation for peers on the semantic web. *Proceedings of Workshop on Secure Data Management in a Connected World*, vol. 3178, pp. 118-132.

Resnick, P.; Zeckhauser, R. (2002): Trust among strangers in Internet transactions: empirical analysis of eBay's reputation systems. *Economics of the Internet and E-Commerce*, vol. 1, pp. 127-157.

Siddiqa, A.; Hashem, I. A. T.; Yaqoob, I.; Marjani, M.; Shamshirband, S. et al. (2016): A survey of big data management: taxonomy and state-of-the-art. *Journal of Network and Computer Applications*, vol. 71, no. 8, pp. 151-166.

Uschold, M. (1996): Building ontologies: towards a unified methodology. *Proceedings of Expert System'96, the 16th Annual Conference of the British Computer Society Specialist Group on Expert Systems*, pp. 1-20.

Wu, J.; Wu, C. H.; Li, Y.; Deng, S. G. (2005): Web service discovery based on ontology and words semantic similarity. *Chinese Journal of Computer*, vol. 28, no. 4, pp. 2054-2062.

Xiong, L.; Liu, L. (2004): PeerTrust: supporting reputation-based trust for peer-to-peer electronic communities. *IEEE Transactions on Knowledge and Data Engineering*, vol. 16, no. 7, pp. 843-857.

Yu, B.; Singh, M, P. (2002): An evidential model of distributed reputation management. *Proceedings of the First International Joint Conference on Autonomous Agents and Multiagent Systems*, pp. 294-301.

Zacharia, G.; Moukas, A.; Maes, P. (2000): Collaborative reputation mechanisms in electronic marketplaces. *Decision Support Systems*, vol. 29, no. 4, pp. 371-388.

Zhang, Y.; Cheng, H. J.; Jiang, X. H.; Sheng, **H.; Yu, T.** (2008): Trust management research survey in E-Commerce. *Chinese Journal of Electronics*, vol. 36, no. 10, pp. 2011-2019.

Zhou, L. H. (2008): Trust management research survey in e-commerce. *Chinese Journal of Electronics*, vol. 36, no. 10, pp. 2011-2019.