The Third International Conference on Internet and Web Applications and Services

Trust Inference in Web-Based Social Networks using Resistive Networks

Mohsen Taherian, Morteza Amini, Rasool Jalili
Computer Engineering Department
Sharif University of Technology
Email: {taherian@ce., m_amini@ce., jalili@}sharif.edu

***Abstract*— By the immense growth of the Web-Based Social Networks (WBSNs), the role of trust in connecting people together through WBSNs is getting more important than ever. In other words, since the probability of malicious behavior in WBSNs is increasing, it is necessary to evaluate the reliability of a person before trying to communicate with. Hence, it is desirable to find out how much a person should trust another one in a network. The approach to answer this question is usually called trust inference. In this paper, we propose a new trust inference algorithm (Called *RN-Trust*) based on the resistive networks concept. The algorithm, in addition to being simple, resolves some problems of previously proposed approaches. The analysis of the algorithm demonstrates that RN-Trust calculates the trust values more accurately than previous approaches.**

I. Introduction

Nowadays, the size of World Wide Web (WWW) is growing rapidly. According to this fast growth, many new facilities are provided to help users establish their interactions with the web. Online service provision and web-based social networks are such facilities. Although transactions and communications in the web are easier and cheaper, determining the trustworthiness of the remote party is not simple at all. When we have not enough information about the other party of our connection, good results of this collaboration can not be guaranteed. In this situation, trust plays a crucial role in connecting people and beginning transactions.

Web-base social networks (WBSNs) are those web sites trying to simulate real social networks on the web. Analyzing the structure of social networks and relationships among people in these networks can provide very valuable information for many research areas such as disease growth. Using the web for simulating social networks, helps researchers extract useful information faster and cheaper and analyze the structure of these kinds of information more accurately. Golbeck completely described the properties of WBSNs in her PhD dissertation [1].

A graph structure is usually used to model trust relationships of people in WBSNs. In this graph, called *trust network*, each node represents a person and each edge represents the trust relationship. A label is assigned to each edge to indicate the trust value of the relationship. In other words, trust value on an edge shows how much one node trusts another one. An important problem of trust in WBSNs is to determine how much one person in the trust network should trust another one who is not directly connected to him. The approach to handle this problem is usually called *trust inference*. The node that represents the person who wants to compute his trust value to another one is called *source*. On the other hand, the node that source wants to infer about is called *sink* or *target*. Trust in WBSNs has some properties such as transitivity, composability, and asymmetry which are described in [1] in detail. Considering the asymmetric property of trust, the trust network would be a directed graph.

Trust algorithms can be divided into two categories; *global* and *local* algorithms. Global algorithms try to compute a universal trust value for each person in the trust network. This trust value is called *reputation*. In this case, regardless of who is asking for a trust recommendation, the same answer is given back. On the other hand, local trust algorithms calculate trust values from the perspective of the person asking for the trust recommendation. Similar to many previous works, we differentiate the concept of trust from the reputation. The main difference is that unlike reputation, trust is subjective. In this paper, from the trust inference algorithm, we mean the local one.

Till now, many trust models have been proposed with different inference algorithms. However, most of these algorithms are suffering from some common problems. Inaccuracy of inferring trust through a chain (a single path from source to sink), ignoring some useful information by eliminating some paths, and the complexity of the inference algorithm are such problems.

In this paper, we propose a new trust inference algorithm in WBSNs called *RN-Trust*. In addition to being simple, the algorithm resolves many problems of the previous ones. Basically, the main idea is to use the *Resistive Network* concept to simulate trust networks. A resistive network is a collection of resistors which are connected in series and parallel. The basic rules of a resistive network could be found in [13].

In the proposed model, each node in the trust network is mapped to a node in the resistive network. Also, for each two adjacent nodes in the trust network, a resistor is placed between their corresponding nodes in the resistive network. It is clear that the resistors' values must have a reverse relation with the trust values. After constructing this resistive network between the source and the sink, the equivalent resistance of this electric circuit can be used as a measure to calculate the

This research is partially supported by Iran Telecommunication Research Center (ITRC).

trust value from the source to the sink. In RN-Trust, there is no need to ignore any information. The problem of inferring trust through a chain is resolved too. In addition, the algorithm is very simple and its time complexity is polynomial. Actually, it requires $O(V^3)$ time in the worst case in which V is the number of nodes in the trust network.

The rest of the paper is organized as follows; In section II, we introduce previous trust models proposed for WBSNs. Our new model and its inference algorithm are discussed in section III. Section IV is devoted to analyze the results and to compare them with those obtained using other algorithms. Finally, we conclude the paper in section V.

II. Related Work

Determining trust value for an unknown entity is a famous problem in distributed environments, even if it would not be directly related to the social networks. Here, we explain previous works on calculating trust values specially those in the social networks area.

A. *EigenTrust*

The EigenTrust algorithm [7] which is proposed to inferring trust in P2P networks, considers trust as a function of corrupt vs. valid files that the node provides. A peer maintains information about the trustworthiness of peers with which it has interacted based on the proportion of good files it has received from that peer. For one peer to determine the trustworthiness of another with which it has not interacted, it needs to gather information from the network and infer the trustworthiness. A peer creates a direct trust rating for another peer based on its historical performance. In its simple form, the algorithm uses a matrix representation of the trust values within the system and over a series of iterations it converges to a globally accepted trust rating of each peer.

However, There is a fundamental difference between trust in P2P networks and trust in social networks. P2P trust is based on the reliability of a node to adhere to absolute correct parameters. A file is either corrupted or it is not. There is not a ”sort of corrupted” file. A node properly implements a protocol or it does not. Again, there is no in-between. In social networks, though, trust is not based on this absolute truth. Two people may hold vastly different opinions about a topic (look only to religion or politics for extreme examples), and there is no absolute truth to determine which one should be trusted and which one should not. A person decides how much to trust another based on personal opinion.

One problem that arises in algorithms that are based on finding the principal eigenvector, like Kamvar (2003) [7], Zeigler and Lausen(2004) [16], and Richardson et al., (2003) [12], is that trust must first be normalized to work within the matrix. This means that the normalized trust value from a person who has made many trust ratings will be lower than if only one or two people had been rated. However, socially, trust is not a finite resource; it is possible to have very high trust for a large number of people, and that trust is not any weaker than the trust held by a person who only trusts one or two others.

B. *TidalTrust Algorithm*

The *TidalTrust* algorithm is proposed by Golbeck in 2005 [1], [2]. An important assumption, Golbeck considered in her algorithm, is that there is no distrust in the trust networks. This means that any trust value is considered as a positive opinion. She claimed that with regard to statistics extracted from the real social networks, this assumption is not unreal.

TidalTrust algorithm considers the trust values to be numbers in a continuous range of $[0..10]$. As mentioned before, a directed graph is used to represent trust relationships. Each edge has a label in the range of $[0..10]$ which 10 means full trust and 0 means no information. Suppose that the node s in the trust network wants to compute its trust value to the node t which is not directly connected to s. First, s sends a request to all its neighbors. This request is recursively forwarded until it reaches to nodes having an edge to t. Then, the trust values of these nodes to t are moved backward across the paths that their corresponding requests are came from. In the backward path, when a node receives more than one value, it uses WAO (weighted average operator) to combine these trust values. This scenario continues until trust values reach to s across the same paths of sending requests but in reverse direction. After this, s uses WAO operator like the other nodes to combine its received trust values and compute final trust value from s to t.

Studying the structure of real world social networks and their properties such as high connectivity, Golbeck applied two restrictions on her algorithm. First, she showed that trust values inferred through shorter paths may be more accurate, so she only considered shortest paths from source to sink in her inference algorithm. Second, she extracted from her analysis that the most trustworthy information usually comes from highest trusted neighbors and lower trusted ones give lower trustworthy information. Thus, she computed a trust threshold for trust network in her algorithm and applied this threshold in combining trust values. This means that in the combination process for a node, those neighbors are considered which that node has a trust value more than this threshold to them. She showed that these restrictions lead to more accurate results in many cases.

Perhaps, the most important preference of Golbeck's algorithm with respect to other ones is its simplicity and its low time complexity ($O(V + E)$). Although she has used a simple operator to combine trust values, looking at the results of other algorithms, we can see that Golbeck's results are better in many cases. This is mainly because of her applied restrictions in the algorithm. TidalTrust algorithm is known as a famous and highly cited algorithm for inferring trust. This is why we chose TidalTrust algorithm as the basis of our results comparison.

Despite of popularity of the TidalTrust algorithm, it has some problems. First, we must mention that with Golbeck's restriction on the length of inference paths, some useful information may be missed. Yet, the more important problem is about single paths (chains) between the source (s) and the sink (t). Suppose that there is only a long chain between the source and the sink, and all nodes in this chain have trust value 9 (high trust) to their neighbors. Now, suppose that there is another chain with the same length with this difference that all nodes have trust value 10 (full trust) to their neighbors except the one just before the sink which has trust value 9 to the sink. With the TidalTrust algorithm there is no difference between these two cases. In both cases, the computed trust value from s to t is 9. However, it is clear that in the former case the inferred trust value must be smaller than the latter one. In fact, increasing the chain length should have a reverse relation with the final inferred trust.

C. Trust Network Analysis with Subjective Logic

Another work on inferring trust value which has used a new logic called Subjective Logic is proposed by Jϕsang in [4]. He claimed that existing logics are not appropriate enough to model uncertain probabilities, so he proposed Subjective Logic and defined its operators. This logic is based on Dempster-Shafer belief combination rule [15].

Unlike the TidalTrust algorithm, the belief combination model of Dempster-Shafer and the consensus operator of Jϕsang assume equally reliable sources. Although the time complexity of Jϕsang algorithm is worse than TidalTrust algorithm, assuming equally reliable sources is its main drawback. We believe in the recommendations of an entity as long as we trust it. The recommenders trust degree is important for evaluating the reliability of the sources. The other point is that WAO operator is simpler than consensus operator, so TidalTrust algorithm is usually preferred over Jϕsang algorithm in the aspect of simplicity in addition to the time complexity. Perhaps, the only significant preference of Jϕsang algorithm over TidalTrust algorithm is the consideration of the concept of distrust which has been ignored by Golbeck.

D. Inferring Trust using Fuzzy Logic

Using fuzzy logic and its operators in trust models has been considered in many works [3], [11], [14]. However, in the area of social networks, it is firstly proposed by Lesani in [8], [9]. His fuzzy algorithm supports the linguistic terms as trust rating of a node to another one in the trust network. The fuzzy membership functions for the linguistic terms such as low, medium, medium low, medium high and high can be defined.

The results of Lesani's simulation indicate that the fuzzy algorithm offers more precise information than the TidalTrust. However, because of that he uses TidalTrust algorithm as the basis of his fuzzy algorithm, it has the same drawbacks of TidalTrust algorithm.

III. The Trust Model

In this section, we introduce our trust model and discuss how this model can resolve problems of the previous models. Basically, the main idea is to use the *Resistive Network* concept to simulate the trust network. To this aim, each trust relationship between two persons is modeled by a resistor such that the more the trust value, the less the value of corresponding resistor. Thus, the trust network is converted to a *Resistive Networks*. To map a trust network to a resistive network we need to define a mapping function that takes a trust label as an input and gives the equivalent resistance as the output. This function and its properties are described later in this paper.

The idea is intuitively clear. Suppose that we have a simple trust network with only two nodes u and v which the trust value from u to v is t. Mapping this trust network to a resistive network, and connect a voltage source between u and v in the resistive network, a current will be flowed between u and v. This current can be interpreted as the trust relation from u to v. If there was no resistor between u and v, u and v would have the same voltage. We can say: u has the same opinion of v. In this situation, the maximum current flows between u and v. If there was a resistor between u and v, then the amount of current flows from u to v decreases. Thus, the trust values have reverse relation with the values of the resistors. The more the trust values, the less the values of resistors. If there is not full trust between two nodes in the trust network, no resistor might be placed between corresponding nodes in the resistive network. On the other hand, if we have low trust between two nodes in the trust network, we must consider a resistor with relatively big resistance between the related two nodes in the corresponding resistive network.

The trust values in our model are supposed to be continuous values in the range of $[0, 1]$. Now, we can define the mapping function between trust values and resistance values. As mentioned in the previous part, trust values should have reverse relation with quantity of resistance. We choose a logarithmic function to this aim which is defined as follows:

$$r = -\log t \tag{1}$$

where r is the resistance value of the resistor assigned to the trust relation which has the value t. It is obvious that when our trust value is 1 which means full trust, the corresponding resistor has zero resistance which means there is no resistor between two nodes.

Similar to other trust models in this paper, the trust network is modeled as a directed graph G. Nodes in G represent people and an edge between the nodes u and v represents a trust relationship between them. A label on the edge (u, v) represents the trust value from u to v. Notice that we assume trust as a asymmetric relation and the trust value from u to v is not the same as the trust value from v to u. To apply the asymmetric property of trust in our modeling, an *ideal diode* is used. That is, an ideal diode is added beside the resistor between u and v. As you know, if there exists an ideal diode between two nodes u and v, it impacts the current to flow in our desired direction. In other words, if a diode is used such that its *cathode* be at the u side and its *anode* be at the v side, only when the current flows from u to v, the resistor plays its role and in reverse direction, diode becomes cut-off and does

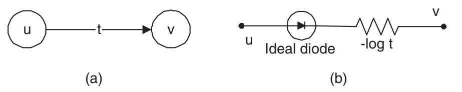

Fig. 1. (a) An edge in a trust network with the trust value t from u to v; (b) Corresponding edge in the resistive network.

not allow the current to flow. The overall process of mapping an edge in the trust network to a part of resistive network is shown in Fig. 1.

A. RN-Trust Algorithm

The aim of this part is to show how we can infer the trust value from s as a source to t as a target that are not directly connected in the trust network G. Here, a new trust inference algorithm is proposed called *RN-Trust*.

To infer the trust value from s to t, the trust network G is mapped to a resistive network, called Res. Note that there is no connection in the resistive network between the nodes which their trust relationship has value 0. The resistance of each resistor can be calculated according to the mapping function introduced before. Given t as the trust value between u and v in G, the corresponding resistor with resistance r between u and v in Res is calculated as $r = -\log^t$.

To calculate the inferred trust value between the source node s and the target node t, first the equivalent resistance between s and t is computed. Suppose that the equivalent resistance between s and t is $R_{eq}(s,t)$. The inferred trust value between s and t, called $T(s,t)$, can be calculated from the following equation:

$$T(s,t) = 10^{-R_{eq}(s,t)} \tag{2}$$

To obtain the equivalent resistance of a resistive network, first an optional voltage difference will be applied between source node and target node and then one of the circuit analysis methods will be used to calculate the amount of electric current that flows through the circuit. Because of the existence of diodes beside the resistors, the current can just flow from s to t. If the applied voltage difference be V and the calculated amount of current be I, the equivalent resistance R can be calculated as follows:

$$R = \frac{V}{I} \tag{3}$$

Among the existing methods for analyzing an electric circuit, the most general one is the *node voltage* method [13]. The second method of circuit analysis, which is so popular, employs *mesh currents* [13] as the independent variables. The idea is to write the appropriate number of independent equations, using mesh currents as the independent variables.

B. Properties of the Model

Our model should obey some principle. Accompanying the introduction of these principles, the way that RN-Trust satisfies them are described as follows.

- The values of all resistors must be greater than or equal to zero. Since a value in range of $[0,1]$ is used as a trust value t and $\log t \leq 0$, regarding to our mapping function $r = -\log t$, we have $r \geq 0$ for all resistors.
- The inferred trust between source and target can not be greater than 1. It is clear that the inferred trust value is always less than or equal to 1. We know that always $R_{eq} \geq 0$, so $10^{-R_{eq}} \leq 1$.
- If there is no path between two nodes, the inferred trust value must be 0. To show why this is true in the proposed method, it is enough to note that if no path is found from a source to a target, an *infinite resistor* is placed between them, so the inferred trust value becomes zero. An infinite resistor can be defined as a resistor with a high resistance capacity, for example $10^3 K\Omega$.
- If all paths from the node s to the node t pass through a node u, then the trust value which s has to u, must be greater than or equal to the trust value which s has to t. To show the correctness of this one, first equivalent resistance between s and u, $R_{eq}(s,u)$, and then between u and t, $R_{eq}(t,u)$, are calculated . Because of that all paths from s to t pass through u, the equivalent resistance between s and t, $R_{eq}(s,t)$, can be considered as the combination of two series resistors $R_{eq}(s,u)$ and $R_{eq}(u,t)$ which are serially connected. Thus, we have:

  $$R_{eq}(s,t) = R_{eq}(s,u) + R_{eq}(u,t)$$

  Because all resistors and equivalent resistors have value greater than or equal to zero, we have:

  $$R_{eq}(s,t) \geq R_{eq}(s,u)$$

  According to our mapping function we can conclude:

  $$10^{-R_{eq}(s,t)} \leq 10^{-R_{eq}(s,u)} \Rightarrow T(s,u) \geq T(s,t)$$

- By increasing a label on the graph G which means increasing the trust value between corresponding nodes, it must not cause to decrease the inferred trust value between any nodes and vice versa. The correctness of direct part is showed. The proof of the reverse part is similar. Since the value of the equivalent resistor has reverse relation with the trust value, increasing the label of the edge (u,v) leads to decrease of the equivalent resistance between those two nodes in Res. It is obvious that when a resistance decreases in the resistive network, the equivalent resistance of the whole network decreases too. Thus, according to the the Eq. 2, the inferred trust value increases.
- Whenever a new path is created from a source to a target which has no intersection with previous paths, the trust value from the source s to the target t must be increased. This fact is intuitively understandable. When a new evidence is obtained, more trust is achieved. We show that RN-Trust satisfy this principle too. If a new disjoint path is created in the trust network, a new disjoint

path will be added to the corresponding resistive network as well. Suppose that the equivalent resistance before adding the new path is $R_{old}(s,t)$, and the equivalent resistance of the new path is $R_{path}(s,t)$. The equivalent resistance between s and t, i.e. $R_{eq}(s,t)$, is equivalent to the combination of two parallel connected resistors $R_{old}(s,t)$ and $R_{path}(s,t)$.

$$R_{eq}(s,t) = \frac{R_{old}(s,t) \times R_{path}(s,t)}{R_{old}(s,t) + R_{path}(s,t)}$$

From the above equation, it can be concluded that:

$$R_{eq}(s,t) \leq R_{old}(s,t)$$

Because of that trust value is proportioned to the reverse of resistors' values, it it obtained that:

$$T(s,t) \geq T_{old}(s,t)$$

Now, we show that RN-Trust algorithm tackles afore mentioned problems of previous algorithms. The first problem pointed in the Jϕsang algorithm is considering equally reliable sources, while in the RN-Trust algorithm, the trust value from each node to its neighbors affect the final inferred value of trust. In the resistive network Res, there are smaller resistors between a node and its more reliable neighbors than its other neighbors. This action causes a little voltage difference between the source node and its reliable neighbors. The second problem is the problem of long chains, which both TidalTrust and its fuzzy algorithm suffer from. In RN-Trust, if there exist a chain, the equivalent resistance of Res will be equivalent to the sum of some series resistors. When the length of the chain is increases, the equivalent resistance is also increases. Thus, the inferred trust value decreases. Another point is that, unlike TidalTrust algorithm, in RN-Trust, no path and no information is missed. In RN-Trust algorithm, all paths, and not only the shortest paths, are considered to calculate the final inferred trust value.

C. Time Complexity of RN-Trust

The first part of our algorithm, RN-Trust, maps the trust network G to the resistive network Res. This mapping process requires $O(E)$ time in which E is the number of edges in G. The main part of the algorithm is to calculate the equivalent resistance R_{eq} in Res.

In order to calculate the equivalent resistance in a resistive network, one of the afore mentioned methods like mesh current analysis is used. This method requires $O(V^3)$ time to complete in which V is the number of nodes in G. Thus, the aggregate time of our algorithm is polynomial, i.e. $O(V^3)$.

As you see, the time complexity of RN-Trust does not significantly differ from other mentioned algorithms such as TidalTrust which takes $O(V+E)$ [1] and Jϕsang's algorithm which takes $O(V+E+L*M)$ (L is the average number of edges exist in the paths between a source and a target, and M is the average number of these paths) [6].

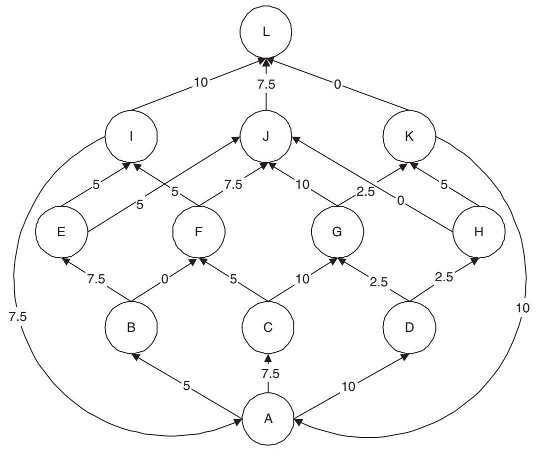

Fig. 2. A Sample trust network for TidalTrust algorithm

TABLE I
RESULTS OF THE TIDALTRUST ALGORITHM FOR THE TRUST NETWORK IN FIG. 2.

	A	B	C	D	E	F	G	H	I	J	K	L
A	-	5	7.5	10	7.5	5	2.5	2.5	5	10	2.5	7.5
B	7.5	-	7.5	10	7.5	0	5.7	2.5	5	5	2.5	8.8
C	7.5	5	-	10	7.5	5	10	2.5	5	10	2.5	7.5
D	10	5	7.5	-	7.5	3	2.5	2.5	5	5	3.8	3
E	7.5	5	7.5	10	-	3	5.7	2.5	5	5	2.5	8.8
F	7.5	5	7.5	10	7.5	-	5.7	2.5	5	7.5	2.5	7.5
G	10	5	7.5	10	7.5	3	-	2.5	5	10	2.5	7.5
H	10	5	7.5	10	7.5	3	5.7	-	5	0	5	0
I	7.5	5	7.5	10	7.5	5	5.7	2.5	-	10	2.5	10
J	-	-	-	-	-	-	-	-	-	-	-	7.5
K	10	5	7.5	10	7.5	5	2.5	2.5	5	10	-	0
L	-	-	-	-	-	-	-	-	-	-	-	-

IV. EXPERIMENTAL RESULTS

In this section, both TidalTrust algorithm and RN-Trust algorithm are applied on a sample trust network shown in Fig. 2. The results of applying TidalTrust algorithm and RN-Trust algorithm are shown in Table I and Table II respectively. To apply RN-Trust algorithm, the trust values are scaled down to the range of $[0,1]$. Also, the outputs are scaled up to the range of $[1..10]$ for better comparison.

To compare RN-Trust's results with the results of TidalTrust algorithm, the trust values from some sources to some targets are analyzed and it is discussed which results are more reasonable with regard to properties of trust in the real social networks.

As already mentioned, the TidalTrust algorithm has a major disadvantage on calculating trust through a chain. For example, consider the trust value from A to E. It is obvious that there is only one path from A to E which is $A \rightarrow B \rightarrow E$. The TidalTrust algorithm gives value 7.5 for the trust of A to E. However, regarding to the trust values in this path, at first blush, trust value from A to E seems to be less than the calculated value by TidalTrust algorithm. When A has not full trust to B, how it can obtain the same trust of B to E? RN-Trust

TABLE II

RN-TRUST TRUST VALUES SCALED UP TO RANGE OF [1..10].

	A	**B**	**C**	**D**	**E**	**F**	**G**	**H**	**I**	**J**	**K**	**L**
A	-	5	7.5	10	3.7	3.7	7.9	2.5	4.3	8.1	4	6.9
B	2.8	-	2.1	2.8	7.5	0	2.2	0.7	3.7	4.9	1.1	5
C	4.7	2.3	-	4.7	1.8	5	10	1.2	3.8	10	3.6	8
D	3	1.5	2.3	-	1.1	1.1	2.5	2.5	1.3	4.2	3	3.5
E	3.7	1.9	2.8	3.7	-	1.4	2.9	0.9	5	5	1.5	6.7
F	3.7	1.9	2.8	3.7	1.4	-	2.9	0.9	5	7.5	1.5	7.3
G	2.5	1.2	1.9	2.5	0.9	0.9	-	0.6	1.1	10	2.5	7.7
H	5	2.5	3.7	5	1.9	1.9	3.9	-	2.2	0	5	3.5
I	7.5	3.7	5.6	7.5	2.8	2.8	5.9	1.9	-	6	3	10
J	-	-	-	-	-	-	-	-	-	-	-	7.5
K	10	5	7.5	10	3.7	3.7	7.9	2.5	4.3	8.1	-	0
L	-	-	-	-	-	-	-	-	-	-	-	-

gives the value 3.7 for the trust from A to E which seems to be a more acceptable result. In fact, in RN-Trust algorithm, the final trust value for a chain is calculated by multiplying all the trust values in the chain. This multiplication comes from the properties of logarithmic function chosen as our mapping function. When there is a chain in the trust network, the equivalent resistor's value of related part in the resistive network will be equal to the sum of all resistors' values in the chain (the resistors in the chain are series resistors).

The second drawback mentioned for TidalTrust algorithm is excluding longer paths in the calculating indirect trust value. This way, some useful information may be lost since shortest paths are only considered. This can be seen from the results of the TidalTrust algorithm in the Table I. For example, the result of TidalTrust gives value 0 for the trust from H to L, because it only considers the path $H \rightarrow K \rightarrow L$ as the only shortest path. However, the question is why the other path $H \rightarrow K \rightarrow A \rightarrow C \rightarrow G \rightarrow J \rightarrow L$ has been ignored. Actually, the trust values across this long path are relatively high and they can affect the entire trust value from H to A. In RN-Trust, the trust value from H to A is 3.5 which shows that the other paths has been also considered. In fact, parallel paths cause parallel resistors in our resistive network and consequently, this make the final trust value become greater.

V. CONCLUSIONS

In this paper, we have introduced a new method to model trust relationships among people. Accompanying this method, a new trust inference algorithm is proposed, called RN-Trust. A trust network is modeled with a resistive circuit. That is, each trust relationship is mapped to a resistor beside an ideal diode. The role of diode is satisfying the asymmetric property of trust in the real world. A logarithmic function is used to map trust values to resistors' values. Furthermore, some required properties of the model were introduced and it was investigated how RN-Trust satisfies them. We discussed how trust inference with RN-Trust takes out some major problems of previous algorithms. Also, the time complexity of RN-Trust algorithm is calculated and proved to be a polynomial time which has not significant difference with previous algorithms.

Future works includes adding more aspects of trust such as *distrust* to the trust network and finding a new algorithm to model this concept (by adding extra electronic elements to our resistive network). Although we proposed this model for WBSNs, our model has enough generality to being applied in other environments. Applying RN-Trust algorithm in new computational environments such as peer-to-peer networks and multi-agent systems is also in our future plans.

REFERENCES

[1] G. A. Golbeck. *Computing and Applying Trust in Web-Based Social Networks*. PhD thesis, University of Maryland, 2005.

[2] G. A. Golbeck and H. James. Inferring binary trust relationships in web-based social networks. *ACM Transactions on Internet Technology*, 6(4):497–529, 2005.

[3] N. Griffiths, K. M. Chao, and M. Younas. Fuzzy trust for peer-to-peer systems. In *P2P Data and Knowledge Sharing Workshop (P2P/DAKS 2006), at the 26th International Conference on Distributed Computing Systems (ICDCS 2006)*, pages 73–73, Lisbon, Portugal, 2006. IEEE Computer Society.

[4] A. Josang. A logic for uncertain probabilities. *International Journal of Uncertainty, Fuzziness and Knowledge-Based Systems*, 9(3):279–311, 2001.

[5] A. Josang. The consensus operator for combining beliefs. *Artificial Intelligence Journal*, 142(1-2):157–170, 2002.

[6] A. Josang, R. Hayward, and S. Pope. Trust network analysis with subjective logic. In *Australasian Computer Science Conference (ACSC2006)*, pages 85–94, Hobart, Australia, 2006.

[7] S. Kamvar, D. Mario, T. Schlosser, and H. Garcia. The eigentrust algorithm for reputation management in p2p networks. In *12th International World Wide Web Conference*, Budapest, Hungary, 2003.

[8] M. Lesani and S. Bagheri. Applying and inferring fuzzy trust in semantic web social networks. In *Canadian Semantic Web Working Symposium*, pages 23–43, 2006.

[9] M. Lesani, S. Bagheri, and H. Abolhassani. Fuzzy trust and combining information, a blueprint for the semantic web trust layer. In *Joint 3rd International Conference on Soft Computing and Intelligent Systems and 7th International Symposium on advanced Intelligent Systems (SCIS & ISIS 2006).*, Tokyo, Japan, 2006.

[10] R. Levin and A. Aiken. Attack resistant trust metrics for public key certification. In *7th USENIX Security Symposium*, San Antonio, Texas, 1998.

[11] D. W. Manchala. Trust metrics, models and protocols for electronic commerce transactions. In *18th International Conference on Distributed Computing Systems*, pages 312–321, Amsterdam, Netherlands, 1998.

[12] M. Richardson, R. Agrawal, and P. Domingos. Trust management for the semantic web. In *Second International Semantic Web Conference*, Sanibel Island, Florida, 2003.

[13] G. Rizzoni. *Principles and Applications of Electrical Engineering*. McGraw-Hill, 5th edition, 2007.

[14] J. Sabater and C. Sierra. Reputation and social network analysis in multi-agent systems. In *First Int. Joint Conference on Autonomous Agents & Multiagent Systems (AAMAS)*, pages 475 – 482, Bologna, Italy, 2002. ACM Press.

[15] G. Shafer. *A Mathematical Theory of Evidence*. Princeton University Press, 1976.

[16] C. N. Ziegler and G. Lausen. Spreading activation models for trust propagation. In *IEEE International Conference on e-Technology, e-Commerce, and e-Service*, Taipei, Taiwan, 2004.