ARTICLE IN PRESS

Future Generation Computer Systems ▮ (▮▮▮▮) ▮▮▮–▮▮▮

Contents lists available at ScienceDirect

Future Generation Computer Systems

journal homepage: www.elsevier.com/locate/fgcs

Multi-dimensional evidence-based trust management with multi-trusted paths

Guojun Wang [a,*], Jie Wu [b]

[a] School of Information Science and Engineering, Central South University, Changsha, Hunan Province, 410083, PR China

[b] Department of Computer and Information Sciences, Temple University, Philadelphia, PA 19122, USA

ARTICLE INFO

Article history:
Received 27 January 2010
Received in revised form
26 April 2010
Accepted 27 April 2010
Available online xxxx

Keywords:
Trust computation
Personalized trust
Transitive trust
Trusted graph
Triangular norms

ABSTRACT

Trust management is an extensively investigated topic. A lot of trust models and systems have been proposed in the literature. However, a universally agreed trust model is rarely seen due to the fact that trust is essentially subjective and different people may have different views on it. We focus on the personalization of trust in order to catch this subjective nature of trust. We propose a multi-dimensional evidence-based trust management system with multi-trusted paths (*MeTrust* for short) to conduct trust computation on any arbitrarily complex trusted graph. The trust computation in *MeTrust* is conducted at three tiers, namely, the node tier, the path tier, and the graph tier. At the node tier, we consider multi-dimensional trust. Users can define a primary dimension and alternative dimensions on their own and users can make their own privileged strategies and setup weights for different dimensions for trust computation. At the path tier, we propose to use the Frank t-norm for users to control the decay rate for trust combination, which can be tuned in between the minimum trust combination (there is no decay in terms of the path length) and the product trust combination (the decay is too fast when the path length is relatively large). At the graph tier, we propose *GraphReduce*, *GraphAdjust*, and *WeightedAverage* algorithms to simplify any arbitrarily complex trusted graph. We employ trust truncation and trust equivalence to guarantee that every link in the graph will be used exactly once for trust computation. We evaluated trust truncation ratio and trust success ratio through extensive experiments, which can serve as a guide for users to select from a wide spectrum of trust parameters for trust computation.

1. Introduction

In order to meet users' expectations in self-organizing networking applications, i.e., to fulfill users' functionalities efficiently and effectively, trust management has been emerging as an essential complementary to security mechanisms. Self-organizing networks are open, dynamic, and prone to a lot of security threats. Without the protection of security mechanisms, it is impossible to run crucial applications properly. With the security protection enabled, crucial applications may still not be able to run efficiently and effectively due to the fact that some nodes may be selfish (i.e., they do not cooperate) or malicious (i.e., they intend to destroy the system).

In a trust management system, evidence about nodes, such as honest, selfish, and malicious behaviors, is collected to conduct trust evaluation (i.e., trust computation, trust combination) among nodes. Based on trust evaluation, decisions can be made in order to encourage the interactions between honest nodes, punish selfish nodes, and exclude malicious nodes. That is, evidence collection, trust evaluation, and decision making are three major components of a trust management system. Evidence can be one dimensional or multi-dimensional [1].

We are also motivated by limitations in existing works on multi-dimensional trust [2] and transitive trust [3]. Most existing works on multi-dimensional trust either combine multiple dimensions into one overall assessment or deal with each dimension separately. On the other hand, most existing works on transitive trust along a trusted path either use the minimum trust among all the trust values (called the *MIN* method) or use the product of all the trust values (called the *PROD* method). The trust information lost may not be able to be neglected in order to achieve accurate trust computation and evaluation. Moreover, most existing works on transitive trust among a trusted graph either require that the graph contains disjoint paths or deal with only a simple graph correctly, which is an idealized and unrealistic model. In particular, there are rarely any efficient and effective works on the trust model that combine the multi-dimensional evidence and user personalization [4,5].

Being aware that trust is essentially subjective and different people may have different views on it, we focus on the personalization of trust in order to catch this subjective nature of trust. We propose a multi-dimensional evidence-based trust

* Corresponding author.
E-mail addresses: csgjwang@mail.csu.edu.cn, csgjwang@gmail.com (G. Wang), jiewu@temple.edu (J. Wu).

0167-739X/$ – see front matter © 2010 Elsevier B.V. All rights reserved.
doi:10.1016/j.future.2010.04.015

management system with multi-trusted paths (*MeTrust* for short) to conduct trust computation on any arbitrarily complex trusted graph. Our contributions are threefold:

(1) Trust computation with multi-dimensional evidence at each node in a trust network. Users can define a primary dimension and alternative dimensions on their own, and users can setup weights for different dimensions for trust computation. A vector of tuples can describe multi-dimensional trust from multi-dimensional evidence more accurately and reasonably.
(2) Personalized trust computation by different privileged strategies and user-controlled trust decay along a trusted path. We define different privileged strategies for the users to derive privileged trust value from multi-dimensional trust. We also propose to use the Frank t-norm for users to control the decay rate for trust combination, which can be tuned in between *MIN* (there is no decay in terms of the path length) and *PROD* (the decay is too fast when the path length is relatively large). This approach combining the multi-dimension evidence and user personalization can be applicable more broadly than existing multi-dimensional trust management systems or personalized trust assessment frameworks.
(3) Trusted graph simplification with trust truncation and trust equivalence. Any arbitrarily complex trusted graph can be simplified for trust computation using our proposed *GraphReduce*, *GraphAdjust*, and *WeightedAverage* algorithms, guaranteeing that every link in the graph will be used exactly once. We show that the proposed algorithms can achieve high availability in trust evaluation by extensive experiments.

The remainder of this paper is organized as follows: In the next section, we introduce some related works. In Section 3, we overview the proposed *MeTrust* system. In Section 4, we show how to compute trust at the node, the path, and the graph tiers. In Section 5, we conduct experiments on the *MeTrust* computation in comparison with the approach that does not simplify the trusted graph. Finally, we conclude this paper and shed some light on future works in Section 6.

2. Related works

2.1. Global vs. local information-based trust models

There are two ways to deal with global information in a trust model. One way is to have an administration center to collect evidence from all the nodes (users) and then the global trust can be computed in the center and be available to all the users. For example, the online auction system eBay [6] requests buyers and sellers to rate each other after each transaction and all the ratings are stored in a center, and then the overall reputation of a participant is the sum of all the ratings over the last 6 months. Such a global model is very simple to implement, but a user has almost no control in trust evaluation.

Another way is to compute a unique global trust for each user in a distributed way that reflects the experiences of all users in the network with the user, e.g., the EigenTrust reputation system [7]. Such a global model does not need an administration center, but it is difficult to guarantee a fast and secure convergence when computing the global trust. There are some more works, such as Poisonedwater [8] for P2P networks and GTMS [9] for WSNs using the global group-based trust metric, to improve the global trust accuracy and convergence rate.

There are also two ways to deal with local information in a trust model. The conservative way is to use only the first-hand information acquired from direct interactions for trust computation, while the aggressive way is to use both the first-hand and second-hand information, the latter of which is acquired from indirect interactions between nodes. The CONFIDANT protocol adopts a local model with the notion of friends for handling the second-hand information in P2P and mobile ad-hoc networks [10]. Compared to a global model, a local model is more scalable, but it is difficult to differentiate misbehaving nodes from normal nodes.

Besides the global and local models, some also use the hybrid ones that take global and local information in between for trust computation. Existing hybrid models employ a trusted graph based on the notion of transitive trust (see Section 2.3). Our *MeTrust* system also adopts a hybrid model for trust computation.

2.2. Multi-dimensional trust

Multi-dimensional trust is also called trust parameters, trust factors, or trust dimensions. Gefen [1] proposes a three-dimensional trust dealing with integrity, benevolence, and ability in the context of e-commerce, which shows that different dimensions of trust are statistically distinct and have different effects on e-commence. Griffiths [2] provides a mechanism for agents to model various dimensions of trust and combine them with other factors when selecting a cooperative partner. This work does not include recommendation trust (i.e., indirect trust) and does not allow any sharing of trust information. Xiong et al. [11] propose a *PeerTrust* model with three basic trust parameters and two adaptive factors in computing trust of peers, and then define a general trust metric to combine them.

To the best of our knowledge, most existing works either combine all dimensions to infer one overall trust or deal with each dimension separately. Different from existing works, our proposed *MeTrust* system allows each user to select a dimension as a primary dimension and put different weights on different dimensions for trust computation.

2.3. Transitive trust

Transitive trust is very natural in human society. For example, if Alice trusts Bob and Bob trusts Clark, then most likely Alice also trusts Clark. A trust network can be formed based on transitive trust with each link representing the trust relationship between two participants. A trusted graph is a sub-network of a trust network, starting from a trustor, ending with a trustee, and connected by a set of trusted paths. Mui et al. [3] and Theodorakopoulos et al. [12] propose to use a trusted graph with multiple disjoint paths between two unknown participants for trust computation. Requiring a trusted graph consisting of only disjoint paths, however, is too restrictive, especially in large networks.

The trusted graph in Jøsang et al. [13] can be any arbitrarily complex network. They propose a method for simplifying a complex network so that it can be expressed in a series–parallel network and then be computationally analyzed. This solution may lead to loss of trust information. They further propose an *edge splitting* method [14] to address this problem. However, this method is valid only on a simple trust network. It may not be valid on a complex trust network.

Golbeck et al. study trust computation also with any complex trusted graph in social networks [4]. A modified breadth-first search algorithm is used to find the trusted paths in a trusted graph and then the binary trust relationship between the trustor and the trustee is inferred. This approach considers only two trust values, i.e., *trusted* or *not trusted*.

Zuo et al. [15] study a framework for trust evaluation based on a set of trusted chains and a trusted graph. In order to maximize the trust value of the trustee evaluated based on a trusted graph, they propose a notion of *a base trusted chain set*, but they do not develop algorithms to identify the set.

Please cite this article in press as: G. Wang, J. Wu, Multi-dimensional evidence-based trust management with multi-trusted paths, Future Generation Computer Systems (2010), doi:10.1016/j.future.2010.04.015

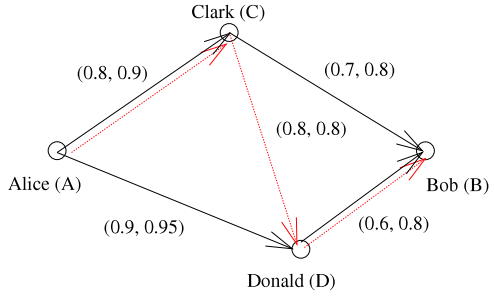

Fig. 1. A simple trusted graph.

2.4. Personalized trust

Huynh [5] proposes a personalized framework for trust assessment. A user can specify how he selects a trust model based on the information about the subject whose trust he needs to evaluate and how the trust model is configured. This is basically a hybrid method that integrates existing trust models, and it is difficult to choose which models to be used in which circumstances and in what ways.

Golbeck et al. identify personalization as one important property of trust [4]. If a user wants an opinion about how much to trust another user, an algorithm that simply averages all recommendations to form an overall assessment may not work well because it reflects the opinion of the whole population. In contrast, they propose a technique to infer the trust based on user similarity.

Our *MeTrust* system shares their vision and we allow each user to setup a trust threshold for trust computation by themselves. Specifically, if a user wants to evaluate whether another user is trustworthy or not, only those recommenders whose trust is no less than the trust threshold can join the trusted graph, and then the trust is evaluated using the trusted graph.

3. MeTrust overview

In the proposed *MeTrust* system, a user Alice ($\mathbb{A}$ for short) wants to contact another unknown user Bob ($\mathbb{B}$ for short) in a self-organizing network. $\mathbb{A}$ first evaluates the trust of $\mathbb{B}$ through a trusted graph, which is formed by finding a set of multiple trusted paths in the network. Fig. 1 shows a simple trusted graph to be used by $\mathbb{A}$ to evaluate the trust of $\mathbb{B}$ through recommenders Clark ($\mathbb{C}$ for short) and Donald ($\mathbb{D}$ for short).

There are 3 trusted paths (*ACB*, *ADB*, and *ACDB*) and 5 links (*AC*, *CB*, *AD*, *DB*, and *CD*) in the graph. A pair of real numbers on the unit interval [0, 1] are associated with each link in the graph. The first number stands for how much a node trusts another node and the second number stands for how much certainty the node trusts the other node with the trust. Generally, the minimum requirement on trusting a recommender is at least 0.5 in order to show more trust than distrust. In *MeTrust*, we consider multi-dimensional trust. The pair of real numbers are inferred from the multi-dimensional trust. For clarity, however, we do not show the multi-dimensional trust in the trusted graph.

In *MeTrust*, the trust computation is conducted at three tiers, namely, the node tier, the path tier, and the graph tier. At the node tier, each node (user) collects the multi-dimensional evidence from its neighbors and then evaluates the trust of its neighbors and the certainty of the trust by combining that evidence. In principle, many existing theories of evidence can be used for trust computation at the node tier. For illustration purposes, we choose to use the Dempster–Shafer Theory (DST) [16]. In particular, the notions of belief and plausibility functions in DST (i.e., lower and upper probabilities) provide a simple way to calculate both the trust and the uncertainty, which will be later used at the path tier and the graph tier.

At the path tier, given a trustor $\mathbb{A}$ and a trustee $\mathbb{B}$, according to $\mathbb{A}$'s requirements, such as the trust threshold (i.e., the minimum requirement on trusting a recommender) and the trust strength (i.e., the number of multi-trusted paths), a set of trusted paths will be found to form a trusted graph. Besides the trustor and the trustee in the graph, all other nodes are recommenders. A trusted path stands for a recommendation trust (i.e., indirect trust) from a trustor to a trustee via some recommenders. We use the Frank t-norm [17] to combine the trust along a trusted path. The decay rate of trust combination can be controlled by the users according to their preferences. To the best of our knowledge, this is the first proposal that uses a controlled decay rate for trust research.

At the graph tier, according to $\mathbb{A}$'s additional requirements, such as trust truncation and trust equivalence, a trusted graph will be simplified in a consistent way such that the trust of each link in the graph will be used exactly once in the process of inferring the final trust of $\mathbb{B}$.

4. MeTrust computation

4.1. MeTrust computation at the node tier

4.1.1. Informal design

At the node tier, each node (user) in the trust network, say, Alice ($\mathbb{A}$ for short) wants to evaluate the trust value concerning each of her neighbors, say, Bob ($\mathbb{B}$ for short). $\mathbb{A}$ first collects the evidence from all her neighbors, and then combines all the evidence together using a certain combination rule in the Dempster–Shafer Theory (DST) [16]. Each neighbor of $\mathbb{A}$ is an independent source who provides the evidence solely based on his own observations without relying on any others.

DST is a mathematical theory of evidence based on belief and plausibility functions. DST is mainly used to combine separate pieces of evidence (information) to calculate the probability of an event. Below, we briefly introduce some basic terms in DST to be used in our multi-dimensional trust.

Frame of discernment (Θ) is represented by an exclusive and exhaustive list of hypotheses, where each hypothesis is a singleton. Its power set, denoted 2^Θ, contains all the subsets of Θ, including the empty set $\emptyset$.

Basic probability assignment (*BPA*) is a mapping $m : 2^\Theta \rightarrow [0, 1]$ that satisfies the following conditions: (1) $m(\emptyset) = 0$ and (2) $\sum_{X\subseteq\Theta} m(X) = 1$, where $0 \leq m(X) \leq 1, \forall X \in 2^\Theta$. The quantity $m(X)$ is a measure of the portion of total belief committed exactly to X, which cannot be further divided among the subsets of X, and does not include any portion of belief committed to any subset of X (excluding X itself). In fact, BPA is a generalization of the probability density function in the traditional probability theory.

Belief function (*Bel*) is a measure of the total amount of belief in X rather than the exact amount committed precisely to X, that is, $Bel(X) = \sum_{Y\subseteq\Theta|Y\subseteq X} m(Y), \forall X \subseteq \Theta$. The Belief function stands for the lower probability of an event.

Plausibility function (*Pl*) is the sum of all the BPAs of the sets that intersect the set X, that is, $Pl(X) = \sum_{Y\subseteq\Theta|Y\cap X\neq\emptyset} m(Y), \forall X \subseteq \Theta$. Plausibility function stands for the upper probability of an event.

We consider that the evidence from each independent source is multi-dimensional. For illustration purposes, we only show two dimensions of the evidence. The first dimension is a frame of discernment denoted $\Theta_F(Honest, Selfish, Malicious)$ or $\Theta_F(H, S, M)$. The second dimension is another frame of discernment, denoted $\Theta_S(Competent, Average, Incompetent)$ or $\Theta_S(C, A, I)$. Θ_F contains 3 singletons, which are $\{H\}$, $\{S\}$, and $\{M\}$, and it contains $2^3 = 8$ subsets, which are $\emptyset$, $\{H\}$, $\{S\}$, $\{M\}$, $\{H, S\}$, $\{S, M\}$, $\{M, H\}$, and

Please cite this article in press as: G. Wang, J. Wu, Multi-dimensional evidence-based trust management with multi-trusted paths, Future Generation Computer Systems (2010), doi:10.1016/j.future.2010.04.015

$\{H, S, M\}$. Among these subsets, we are only interested in 4 subsets, namely, $\{H\}$, $\{S\}$, $\{M\}$, and $\{S, M\}$. We call the last subset *Misbehavior*.

Similarly, Θ_S also contains 3 singletons and 8 subsets, but we are also only interested in 4 subsets, namely, $\{C\}$, $\{A\}$, $\{I\}$, and $\{C, A\}$. We call the last subset *PrettyGood* in the sense that both *competent* and *average* capabilities are pretty good in the eyes of different users. We point out that there isn't any standard criterion in the selection of interested subsets, which are intermediate variables for the preferred subset. We only require that these subsets are reasonable and can be distinguished from each other.

4.1.2. Informal definitions

Multi-dimensional evidence. In the last subsection, we introduced the notion of traditional evidence as the frame of discernment Θ in DST. The multi-dimensional evidence can be viewed as a vector version of the traditional evidence as $\vec{\Theta}$. We assume $N(\geq 1)$ is the total number of dimensions, but it will not appear in the notation for simplicity. Therefore, $\vec{\Theta}_0$ stands for the first dimension of the frames of discernment, $\vec{\Theta}_1$ stands for the second dimension of the frames of discernment, and so on.

Multi-dimensional functions. With the vector version of the frame of discernment Θ, it is easy to define the vector version of basic probability assignment m, belief function *Bel*, and plausibility function *Pl*, which are $\vec{m}$, $\vec{Bel}$, and $\vec{Pl}$, respectively. Similar to $\vec{\Theta}$, we assume $N(\geq 1)$ is the total number of dimensions, but it will not appear in the notations.

Primary dimension. The primary dimension, denoted $\vec{\Theta}_{Pri}$, is a dimension selected from $\vec{\Theta}$, indicating the major concern among all the dimensions when evaluating the trust. Different users can have their own selections on which dimension is to be the primary dimension. Once $\vec{\Theta}_{Pri}$ is determined, all other dimensions become *alternative dimensions*.

Interested subsets. Regarding $\vec{\Theta}_i$ $(0 \leq i \leq N-1)$, if the number of singletons in this frame of discernment is denoted $\vec{n}_i$, where $\vec{n}$ is a vector of such numbers corresponding to $\vec{\Theta}$, then the total number of subsets in the power set $2^{\vec{\Theta}_i}$ is $2^{\vec{n}_i}$. *Interested Subsets*, denoted $\vec{\text{IntSub}}_i$, are those subsets selected from the power set, including the empty set $\emptyset$ and the full set containing all the singletons in $\vec{\Theta}_i$. If no confusion will occur, we also use $\vec{\Theta}_i$ to denote the full set.

Preferred subset. For each of those interested subsets, e.g., $\vec{IntSub}_i$ $(0 \leq i \leq N-1)$, one subset is selected as the *preferred subset*, denoted $\vec{PreSub}_i$, indicating a user will put more preferences on this subset than any other subset for trust evaluation. Different users can have their own selections on which subset to be the preferred subset.

Multi-dimensional trust. The *multi-dimensional trust*, denoted $\vec{Trust}$, is a vector of tuples, each of which is associated with both the belief and the plausibility functions for each preferred subset of the multi-dimensional evidence in question, that is, $\vec{Trust} = (\vec{Bel}(\vec{PreSub}), \vec{Pl}(\vec{PreSub}))$, and $\vec{PreSub}_i$ is used for evaluating $\vec{Bel}_i$ and $\vec{Pl}_i$, where $0 \leq i \leq N-1$ and N is the number of dimensions.

4.1.3. Trust combination functions

Each node evaluates the multi-dimensional trust of each neighbor by collecting and then combining the multi-dimensional evidence from all the neighbors. We use the classical combination rule proposed by Dempster [16], called the *Dempster's combination rule*, although a plethora of alternative combination rules can also be used, such as the robust combination rules (RCR) [18]. For brevity, we omitted introducing the combination rules (functions) due to space limitation.

4.2. MeTrust computation at the path tier

4.2.1. Informal design

At the path tier, we investigate how to combine the multi-dimensional trust along a single trusted path in the trusted graph. Most existing works, such as Sun et al. [19], use the *conjunctive rule*, which states that *concatenation propagation of trust does not increase trust*. We apply this rule at the path tier. We introduce *Triangular norms* to do so in the multi-dimensional setting.

In the Dempster–Shafer theoretical framework [16], the difference between the plausibility and the belief functions measures the uncertainty of trust. The former is always no less than the latter and the result is a real number on the unit interval [0, 1]. This information will be aggregated at the path tier and later be used for trust combination at the graph tier.

4.2.2. Informal definitions

We first introduce *Triangular norms* (*t-norms* for short) [17], an elegant mathematical tool to model the conjunctive behavior. Then, we propose the notion of *privileged trust* in order to apply the conjunctive rule using t-norms.

Triangular norm. A triangular norm (t-norm) is a binary operation T on the unit interval [0, 1], i.e., a function $T : [0, 1]^2 \to [0, 1]$, such that for all $x, y, z \in [0, 1]$, the following four axioms are satisfied: (1) Commutativity: $T(x, y) = T(y, x)$; (2) Associativity: $T(x, T(y, z)) = T(T(x, y), z)$; (3) Monotonicity: $T(x, y) \leq T(x, z)$ whenever $y \leq z$; and (4) Boundary condition: $T(x, 1) = x$.

The following are four t-norms. (1) Minimum t-norm: $T_M(x, y) = \min(x, y)$; (2) Product t-norm: $T_P(x, y) = x \cdot y$; (3) Łukasiewicz t-norm: $T_L(x, y) = \max(x + y - 1, 0)$; and (4) The parameterized family $(T^F_\lambda)_{\lambda \in [0, +\infty]}$ of Frank t-norm is given by

$$T^F_\lambda(x, y) = \begin{cases} T_M(x, y) & \text{if } \lambda = 0 \\ T_P(x, y) & \text{if } \lambda = 1 \\ T_L(x, y) & \text{if } \lambda = +\infty \\ \log_\lambda\left(1 + \dfrac{(\lambda^x - 1)(\lambda^y - 1)}{\lambda - 1}\right) & \text{otherwise.} \end{cases}$$

Privileged trust and uncertainty of trust. In *MeTrust*, the multi-dimensional trust $\vec{Trust}$ is a vector of tuples $(\vec{Bel}(\vec{PreSub}), \vec{Pl}(\vec{PreSub}))$. We provide three privileged strategies for the users to use this metric in a simple fashion, namely, belief privileged strategy (using only $\vec{Bel}(\vec{PreSub})$), plausibility privileged strategy (using only $\vec{Pl}(\vec{PreSub})$), and hybrid privileged strategy (a weighted average using both $\vec{Bel}(\vec{PreSub})$ and $\vec{Pl}(\vec{PreSub})$). Each strategy can be further divided into two sub-strategies, one considering only the primary dimension and the other one considering all the dimensions. We call the overall trust deduced from such strategies *privileged trust*, denoted *pTrust*, which is a real number on the unit interval [0, 1]. Accordingly, we also deduce *uncertainty of trust*, denoted *unc*, from such strategies.

4.2.3. Trust combination functions

Trust combination functions consist of two operations: (1) the $\mathbb{P}$ operation for a user to select a privileged strategy to deduce the privileged trust (*pTrust*) and the uncertainty of trust (*unc*) from a $\vec{Trust}$ (multi-dimensional trust); (2) the $\mathbb{T}$ operation for a user to select a specific t-norm to combine a sequence of multi-dimensional trust values to the final privileged trust and the uncertainty of trust.

The $\mathbb{P}$ operation takes a multi-dimensional trust $\vec{Trust}$, a privileged strategy together with its sub-strategy as input parameters, and returns two real numbers on the unit interval [0, 1] called the privileged trust ($pTrust$) and the uncertainty of trust (*unc*). That is, $(pTrust, unc) = \mathbb{P}(\vec{Trust}, strategy, sub\text{-}strategy)$.

Please cite this article in press as: G. Wang, J. Wu, Multi-dimensional evidence-based trust management with multi-trusted paths, Future Generation Computer Systems (2010), doi:10.1016/j.future.2010.04.015

Algorithm 1 $\mathbb{P}$ operation

Input: ($\vec{Trust}$, *strategy* = "Hybrid",
  sub-strategy = "All", W_{Bel}, W_{Pl}, $\vec{W}$)
Output: t, *unc*
$//\vec{Trust}$ is the tuple ($\vec{Bel}(\vec{PreSub})$, $\vec{Pl}(\vec{PreSub})$);
$//N$ is the total no. of dimensions of the evidence.
$t = 0$; $unc = 0$
for $i = 0$ **to** $N - 1$ **do**{
  $t = t + \vec{W}_i * (W_{Bel} * \vec{Bel}_i(\vec{PreSub}_i) + W_{Pl} * \vec{Pl}_i(\vec{PreSub}_i))$
  $unc = unc + \vec{W}_i * (\vec{Pl}_i(\vec{PreSub}_i) - \vec{Bel}_i(\vec{PreSub}_i))$}

The *strategy* can be "Belief", "Plausibility", or "Hybrid", and the *sub-strategy* can be "Primary" or "All". In the case of the "Hybrid" strategy, two non-negative weights called W_{Bel} and W_{Pl} will be used to combine belief and plausibility functions, satisfying $W_{Bel} + W_{Pl} = 1$. In the case of the "All" sub-strategy, a vector of N non-negative weights $\vec{W}$ satisfying $\sum_{i=0}^{i=N-1} \vec{W}_i = 1$ will be used for trust combination, where N is the total number of dimensions.

Algorithm 1 shows the algorithmic description of the $\mathbb{P}$ operation with the "Hybrid" strategy and the "All" sub-strategy. In fact, this operation is a general case that covers all other strategies and sub-strategies by specifying special combinations of weights, including W_{Bel}, W_{Pl}, and $\vec{W}$. For example, if $W_{Bel} = 1$ and $\vec{W}_i = 1$ (here the ith dimension is the primary dimension), then this operation is with the "Belief" strategy and the "Primary" sub-strategy.

The $\mathbb{T}$ operation is as follows: Due to the property of non-increasing monotonicity of t-norm, we use it to combine privileged trust values propagated along a trusted path, which conforms to the decay nature of trust propagation through recommenders. In *MeTrust*, those conservative users may prefer such decay to be as fast as possible while those aggressive users may prefer it as slow as possible. The parameterized family $(T_\lambda^F)_{\lambda\in[0,+\infty]}$ of Frank t-norm can meet such requirements by setting appropriate parameters. For example, if the user is aggressive and set λ to be 0 in MeTrust, then the minimum t-norm will be used. That is to say, for a trusted path including two edges with trust values x and y, the combined trust value of the trusted path is the minimum value among x and y. So, there will be no decay in terms of the path length. On the contrary, if the user is conservative and set λ to be 1, then the product t-norm is used. Due to the fact that both x and y are not bigger than 1, the product will not be bigger than either x or y. That is to say, the decay will be fast when the path length is relatively large.

Algorithm 2 shows the algorithmic description of the $\mathbb{T}$ operation. The input includes a sequence of $\vec{Trust}$ values along a trusted path, denoted $\vec{PathTrust}$, and some related parameters, such as the privileged strategy together with the sub-strategy, and some weights. The output includes two real numbers on the unit interval [0, 1] (i.e., the final privileged trust and the aggregated uncertainty of trust).

4.3. MeTrust computation at the graph tier

4.3.1. Informal design

Different users may have different requirements on trust strength when requesting others to do a thing, e.g., to build a secure communication between two parties. Compared to using a single trusted path in a trust network for evaluating the trust value from a trustor $\mathbb{A}$ to a trustee $\mathbb{B}$, multi-trusted paths from $\mathbb{A}$ to $\mathbb{B}$ can increase the trust strength perceived by $\mathbb{A}$ on $\mathbb{B}$. The multi-paths from $\mathbb{A}$ to $\mathbb{B}$ make up a trusted graph. If the multi-paths in the graph are node- or link-disjoint with each other (except for the trustor and the trustee), we can evaluate the trust for each path

Algorithm 2 $\mathbb{T}$ operation

Input: ($\vec{PathTrust}$, *strategy*, *sub-strategy*,
  W_{Bel}, W_{Pl}, $\vec{W}$)
Output: t, *unc*
$//L$ is the length of a trusted path in a trusted graph.
$(t, unc) = \mathbb{P}(\vec{PathTrust}_0$, *strategy*, *sub-strategy*,
  W_{Bel}, W_{Pl}, $\vec{W}$)
for $i = 1$ **to** $L - 1$ **do**{
  $(t', unc') = \mathbb{P}(\vec{PathTrust}_i$, *strategy*, *sub-strategy*,
    W_{Bel}, W_{Pl}, $\vec{W}$)
  $t = (T_\lambda^F)(t, t')$; $unc = unc + unc'$}
$unc = unc/L$

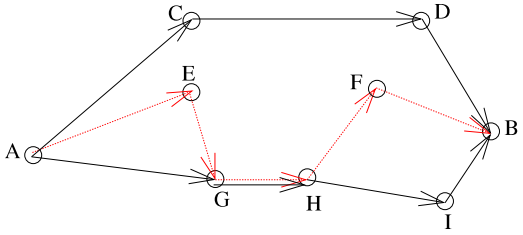

Fig. 2. The impact of a shared link in trust evaluation.

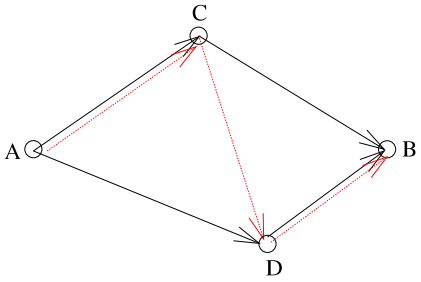

Fig. 3. The impact of a crossing link in trust evaluation.

using t-norm and then combine the trust among the multi-paths by weighted average. The weight of a trusted path is deduced from the uncertainty of trust along the trusted path.

However, if the multi-paths do not satisfy either node disjointness or link disjointness, then some nodes and/or some links in the graph will be shared by at least two trusted paths. If a shared link is used more than once in trust computation, then the combined trust is problematic since the impact of a shared link to trust computation is not clear. Fig. 2 shows this problem with a shared link **GH**, which appears in both paths *AE***GH***FB* and *A***GH***IB*. The general case of a shared link is a *shared segment* consisting of a sequence of consecutive links shared by two or more paths. We use shared link and shared segment interchangeably.

If the trusted graph contains a link that crosses two trusted paths, it is also problematic since the impact of a crossing link to trust computation is also not clear. Fig. 3 shows three trusted paths with a crossing link. The link **CD** in the trusted path *A***CD***B* crosses two paths *ACB* and *ADB*. The general case of a crossing link is a *crossing segment* consisting of a sequence of consecutive links crossing two or more paths. We use crossing link and crossing segment interchangeably.

We propose three algorithms to tackle these problems. *GraphReduce* reduces a trusted graph to a graph containing only a maximum number of node- or link-disjoint multiple paths from a trustor $\mathbb{A}$ to a trustee $\mathbb{B}$. *GraphAdjust* adjusts the reduced graph by considering the impacts of shared/crossing links and all other links not in the reduced graph, which guarantees that each link in a trusted graph will be used exactly once for trust

Please cite this article in press as: G. Wang, J. Wu, Multi-dimensional evidence-based trust management with multi-trusted paths, Future Generation Computer Systems (2010), doi:10.1016/j.future.2010.04.015

Algorithm 3 Weighted Average

Input: ($\vec{GraphTrust}$, *strategy*, *sub-strategy*, $W_{Bel}, W_{Pl}, \vec{W}$)
Output: t
//K is the No. of trusted paths in a trusted graph.
$(t, unc) = \mathbb{T}(\vec{GraphTrust}_0, strategy, sub\text{-}strategy, W_{Bel}, W_{Pl}, \vec{W})$
$cert = 1 - unc$; $t = cert * t$
for $i = 1$ **to** $K - 1$ **do**{
  $(t', unc') = \mathbb{T}(GraphTrust_i, strategy, sub\text{-}strategy, W_{Bel}, W_{Pl}, \vec{W})$
  $t = t + (1 - unc') * t'$; $cert = cert + (1 - unc')$}
$t = t/cert$

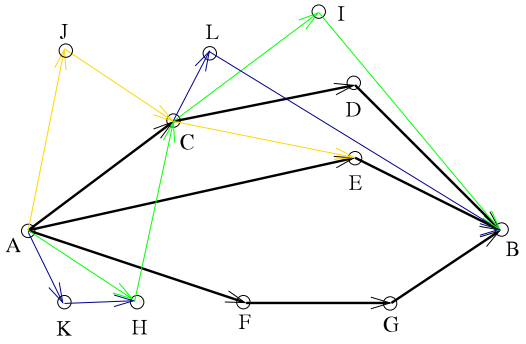

Fig. 4. An example trusted graph.

computation. *WeightedAverage* computes the final privileged trust based on the adjusted reduced graph. *WeightedAverage* is also used to compute some "intermediate" privileged trust for *GraphReduce* and *GraphAdjust* algorithms, and thus we introduce it first.

4.3.2. The WeightedAverage algorithm

Given a trusted graph (or its sub-graph), we propose the *WeightedAverage* algorithm shown in Algorithm 3 to compute the combined trust by averaging all the privileged trust values among all the trusted paths weighted by the certainty of trust along each path. Here, the certainty of trust is defined as 1—the uncertainty of trust. The input of *WeightedAverage* includes $K(\geq 1)$*Path*$\vec{Trust}$ values, denoted $\vec{GraphTrust}$, and some related parameters, such as the privileged strategy together with the sub-strategy, and some weights. The output is a privileged trust, which is a real number on the interval [0, 1].

4.3.3. The GraphReduce algorithm

We assume that a trusted graph consisting of $K(\geq 1)$ trusted paths from a trustor $\mathbb{A}$ to a trustee $\mathbb{B}$ forms a directed acyclic graph (DAG). Fig. 4 shows an example trusted graph with $K(=6)$ trusted paths from $\mathbb{A}$ to $\mathbb{B}$, which are *ACDB*, *AEB*, *AFGB*, *AHCIB*, *AJCEB*, and *AKHCLB*, respectively.

Notice that, if the combined trust using our *WeightedAverage* algorithm for a trusted path drops below the trust threshold (e.g., 0.5), then this path is not qualified for trust combination at the graph level, and thus it will be truncated (i.e., removed) from the trusted graph. This is called *trust truncation* in this paper.

After trust truncation, all the qualified trusted paths in a DAG may not be node- or link-disjoint. We propose *GraphReduce*, shown in Algorithm 4, to find a maximum number of node- or link-disjoint paths by enumerating all possible combinations of all the qualified trusted paths. The input includes $K(\geq 1)$ *Path*$\vec{Trust}$ values, denoted $\vec{GraphTrust}$, and some related parameters, such as the privileged strategy together with the sub-strategy, and some weights. The

Algorithm 4 Graph Reduce

Input: ($\vec{GraphTrust}$, *strategy*, *sub-strategy*, $W_{Bel}, W_{Pl}, \vec{W}$)
Output: REDU, RESI
//K is the no. of trusted paths in a trusted graph.
call $\mathbb{T}$ for each path in $\vec{GraphTrust}$
for $i = K$ **downto** 1 **do**{
  call *WeightedAverage* for each combination of i paths
  if at least one combination is found **then**
    return REDU = i paths with largest weighted average, RESI = The remaining $K - i$ paths
}

output includes the reduced graph called *REDU* (containing all the node- or link-disjoint paths) and the residual graph called *RESI* (containing all the residual paths).

4.3.4. The GraphAdjust algorithm

Refer to the example trusted graph in Fig. 4. Our *GraphReduce* algorithm outputs node-disjoint reduced DAG with 3 paths (*ACDB*, *AEB*, and *AFGB*) or link-disjoint reduced DAG with 4 paths (the other one is *AHCIB*). For brevity, we discuss *GraphAdjust* only with node-disjointness. The case of link-disjointness is similar and we do not discuss it in detail due to space limitation.

GraphAdjust uses several notions for graph simplification. The first notion is *shared segment* (*shared link*). A shared segment is a sequence of consecutive links shared by two or more paths. If a shared segment is shared by both *REDU* and *RESI*, e.g., *EB* is shared by *AEB* and *AJCEB*, it can simply be removed from *RESI*; if shared by two or more paths in *RESI*, e.g., *HC* is shared by *AHCIB* and *AKHCLB*, then we only keep one segment in *RESI*.

The second notion is *crossing segment* (*crossing link*). A crossing segment (CROSS) is a sequence of consecutive links crossing between two or more paths. CROSS is a recommendation segment from a trustor to a trustee. In the trusted graph, we can find one or more recommendation segments (SEGS) from some other recommenders to the trustee. A recommendation segment is a sequence of consecutive links from the trustee backward to a node, which is either the trustor Alice in the whole trusted graph or the first node from which two or more recommendations are made. In order to remove CROSS from the graph, we adjust the trust values in SEGS such that the weighted average trust of the adjusted SEGS is the same as that of the original SEGS together with CROSS. In Fig. 4, *CE* is a CROSS and *AE* is the SEGS, and thus *CE* will be removed and its impact will be reflected on *AE*.

The third notion is *trust equivalence*. A set of node-disjoint paths (segments) from a trustor (one recommender) to a trustee (another recommender) are called SEGS. A SEGS can be simplified to one designated path (PATH) in the SEGS. But, there may be a crossing segment (CROSS) between these paths in the SEGS. So, we can adjust the trust value for all related links in the PATH such that the weighted average of the privileged trust of the SEGS is equal to the combined privileged trust of the adjusted PATH. The adjusted PATH will then be used as an equivalence of the original SEGS. For a large scale trusted graph, there may be many trusted sub-graphs that can also output node-disjoint reduced DAGs. Then, this trust equivalence can be approximated by writing a computer program to make an adjusted PATH using an iterative method, which considers the impact of the uncertainty of trust at each link of the PATH.

The fourth notion is *segment dependence*. We use an example to show its meaning. In Fig. 4, we identify $SEGS_1 = AKH, AH$ (*AH* is the PATH) and $SEGS_2 = AHC, AC, AJC$ (*AC* is the PATH). Here, the simplification of $SEGS_2$ is dependent on that of $SEGS_1$ because

Please cite this article in press as: G. Wang, J. Wu, Multi-dimensional evidence-based trust management with multi-trusted paths, Future Generation Computer Systems (2010), doi:10.1016/j.future.2010.04.015

Algorithm 5 Graph Adjust

1: **Input**: ($Graph\vec{Trust}$, *strategy*, *sub-strategy*,
2: W_{Bel}, W_{Pl}, $\vec{W}$)
3: **Output**: REDU, RESI
4: **call** *GraphReduce* on $Graph\vec{Trust}$ to get
5: *REDU* and *RESI*
6: Remove shared, crossing, disjoint segments
7: from *RESI*
8: }

segment *AH* in $SEGS_1$ is no longer there after $SEGS_2$ is simplified. In this case, we need to first simplify $SEGS_1$ and then simplify $SEGS_2$ due to this segment dependence.

Algorithm 5 (*GraphAdjust*) combines all the above notions. The algorithm first calls *GraphReduce* to get *REDU* and RESI graphs, then removes shared segments, crossing segments, and disjoint segments from *RESI* and adjusts *REDU* accordingly. The algorithm finally outputs the adjusted REDU graph (the trust values on some links will be changed, but the topology of the graph remains unchanged) and the RESI graph (it becomes an empty graph).

5. Experimental evaluation

5.1. Experimental methodology

A trust network can be very large, but a trusted graph deduced from a trust network can be very small according to small-world phenomenon. We directly construct trusted graphs rather than deducing them from a trust network. The *degree of disjointness* of a trusted graph is the ratio of the maximum number of disjoint paths over the total number of trusted paths in the trusted graph. We configure the degree 75% for node-disjointness and 85% for link-disjointness, respectively, when constructing the trusted graphs.

Trust threshold is a real number on the interval [0.5, 1]. We use discrete numbers 0.5, 0.6, 0.7, 0.8, and 0.9, where 0.5 stands for a low trust and 0.9 stands for a high trust.

Trust can be multi-dimensional. Each dimension (i.e., a *frame of discernment* in DST [16]) can have a list of hypotheses from which subsets will be formed. For simplicity, we conduct experiments on one-dimensional trust with belief (*Bel*) and plausibility (*Pl*) functions defined for a preferred subset. We evaluate the trust on the preferred subset, which is similar to traditional overall trust assessment (*Trust*) and its associated confidence level or certainty. We use the hybrid strategy with an equal weight for both *Bel* and *Pl*. That is, $Trust = (Bel + Pl)/2$ and $Certainty = 1 - (Pl - Bel)$.

Each link in a trusted graph is associated with a tuple (*Bel*, *Pl*). For a certain trust threshold (*TH*), we randomly generate a real number on the interval [*TH*, 1], which stands for *Bel*. Then, we randomly generate another real number on the interval [*Bel*, 1], which stands for *Pl*.

Trust strength is represented by the minimum and maximum numbers of trusted paths in a trusted graph. *MinN* is the minimum number of qualified trusted paths after evaluating a trusted graph. *MaxN* is the maximum number of trusted paths when constructing a trusted graph. If the combined trust of a trusted path is below a certain trust threshold, then it will be truncated (i.e., removed) from the trusted graph, and thus it is counted as an unqualified trusted path. After trust truncation, our *GraphReduce* algorithm divides the trusted paths into the maximum number of disjoint paths (i.e., the *REDU* graph) and the residual paths (i.e., the *RESI* graph). All the paths in the *RESI* graph are also counted as unqualified paths.

MaxL is the maximum path length of a trusted path in a trusted graph. Since we assume a trustor does not know a trustee in a trusted graph, the minimum path length (*MinL*) of a trusted path is at least 2. Notice that the total number of trusted paths for each kind of path length from *MinL* to *MaxL* is the same in the probabilistic sense.

We evaluate trust truncation ratio (*TTR*), which is defined as the ratio of the total number of truncated (i.e., removed) paths in all the trusted graphs over the total number of trusted paths in all the trusted graphs.

We also evaluate trust success ratio (*TSR*), which is defined as the ratio of the total number of trusted graphs evaluated as *successful* over the total number of trusted graphs being evaluated. A trusted graph is evaluated as successful if it satisfies two conditions. (1) The overall trust assessment is no less than the trust threshold. (2) The total number of qualified trusted paths in the trusted graph is no less than *MinN*. A trustor can trust a trustee if the trusted graph connecting the trustor and the trustee is evaluated as successful.

5.2. Experimental results

For each group of experiments, we conduct 1000 experimental runs and report their average results. All our experiments are reported over the λ parameter in the Frank t-norm [17]. $\lambda = 0$ means to use the minimum trust among all the trust values along a trusted path (called the *MIN* method). $\lambda = 1$ means to use the product of all the trust values (called the *PROD* method). A real number λ on the interval [0, 1] is used for users to control the decay rate for trust combination, which can be tuned in between *MIN* (there is no decay in terms of the path length) and *PROD* (the decay is too fast when the path length is relatively large).

We report the first group of experiments on trust truncation ratio (*TTR*) over λ. Fig. 5(Left) shows the case of $MaxL = 3$ and Fig. 5(Right) shows the case of $MaxL = 6$. We used the number 6 here for large trust networks according to *Six Degrees of Separation* phenomenon [20]. Accordingly, the number 3 here is used for small trust networks, and thus the trusted graphs are with at most 3 links for each trusted path in a trusted graph. This figure shows that *TTR* increases over λ, *MaxL*, and *TH*. *TTR* serves as an intermediate result. Trust success ratio (*TSR*) serves as the *final* result, which will be reported right after the *TTR* result. Our experimental results show that *TSR* decreases as *TTR* increases.

We report the second group of experiments on trust success ratio (*TSR*) over λ without any graph simplification. Fig. 6 shows the relationship between *TSR* and λ, *MinN*, *MaxN*, *MaxL*, and *TH*. In order to make a trustor perceive enough trust on a trustee, *TSR* needs to be kept relatively high. This figure serves as a guide for users to select a proper combination from a wide spectrum of these trust parameters. Personalized trust can be achieved through adjusting these trust parameters based on users' preferences.

The more conservative users may prefer to choose λ that is closer to 1 in order to decay the trust more quickly, like the *PROD* method, and the more aggressive users may prefer to choose λ that is closer to 0 in order to decay the trust more slowly, like the *MIN* method.

Trust strength is represented with the pair (*MinN*, Max*N*). The values (1, 2) stand for low trust strength and the values (3, 6) stand for high trust strength. When fixing all other parameters, a higher trust strength achieves a higher *TSR*. The cost of a higher trust strength is that more trusted paths need to be found to construct a trusted graph.

MaxL also affects *TSR*. When fixing all other parameters, a larger *MaxL* leads to a lower *TSR* due to the fact that those longer trusted paths will be truncated with a larger probability since the trust will decay along each link of a trusted path.

TH also has some impact on *TSR*. When fixing all other parameters, *TSR* decreases over *TH*. It is easy to understand this trend. Regarding the same path length for two trusted paths with

Please cite this article in press as: G. Wang, J. Wu, Multi-dimensional evidence-based trust management with multi-trusted paths, Future Generation Computer Systems (2010), doi:10.1016/j.future.2010.04.015

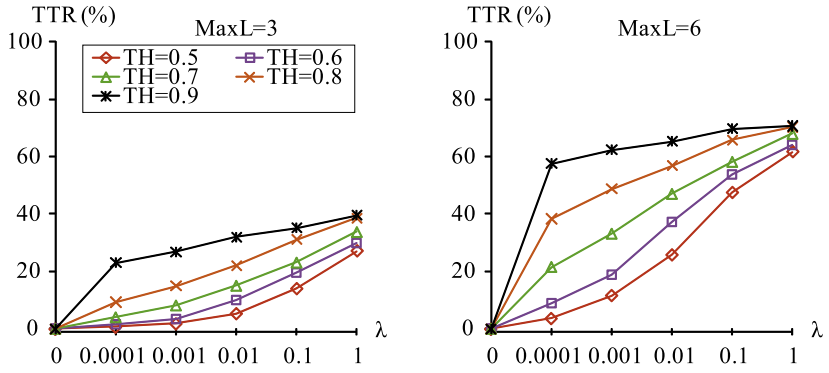

Fig. 5. Trust truncation ratio without graph simplification.

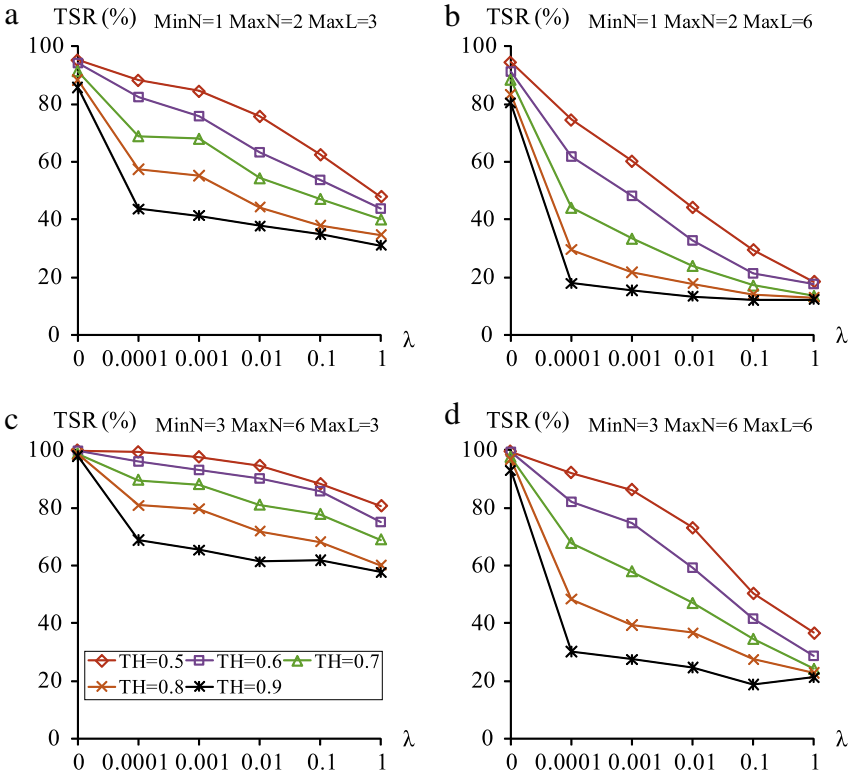

Fig. 6. Trust success ratio without graph simplification.

different *TH* values, it is easier for the combined trust to drop below the higher *TH* value than the lower *TH*.

The above groups of experiments do not consider graph simplification. Below, we consider graph simplification approaches and compare them with the approach that does not simplify the trusted graphs. The two graph simplification approaches are node-disjointness approach, denoted REDU = *NODE*, and link-disjointness approach, denoted REDU = *LINK*. The approach without graph simplification is denoted *REDU* = *NO*.

Our third group of experiments is on trust truncation ratio (*TTR*) over λ comparing the three approaches. Besides those paths truncated by applying trust truncation, those paths in the residual graph (*RESI*) are also counted as truncated paths. Fig. 7 shows the results over different combinations of *MaxL* and *TH*. We consider a low trust threshold (*TH* = 0.5) and a high trust threshold (*TH* = 0.8). It shows that *TTR* increases over λ, *MaxL*, and *TH*. When fixing all other parameters, it is the lowest when *REDU* = *NO*, the highest when *REDU* = *NODE*, and in between when REDU = *LINK* due to two reasons: (1) the node-disjointness is a special case of the link-disjointness, and (2) the link-disjointness is a special case of a generic trusted graph (i.e., the graph without any graph simplification).

Our fourth group of experiments is on trust success ratio (*TSR*) over λ comparing the three approaches. Fig. 8 shows different combinations of *MinN*, *MaxN*, and *MaxL*. In each case, we consider a low trust threshold (*TH* = 0.5) and a high trust threshold (*TH* = 0.8). This figure shows very similar trends over λ, *MinN*, *MaxN*, *MaxL*, and *TH*, as in Fig. 6. Moreover, when fixing all other parameters, there is a slight decrease over the three different approaches in the order of *REDU* = *NO*, *REDU* = LINK, and *REDU* = *NODE*. It means that the approaches with graph simplification can achieve a similar trust success ratio, but a user can perceive stronger trust strength since he is relying on the node- or link-disjointed trusted paths in the trusted graph for trust computation.

6. Conclusion

Trust management has been increasingly attracting more and more attention due to its potentially very broad and highly

Please cite this article in press as: G. Wang, J. Wu, Multi-dimensional evidence-based trust management with multi-trusted paths, Future Generation Computer Systems (2010), doi:10.1016/j.future.2010.04.015

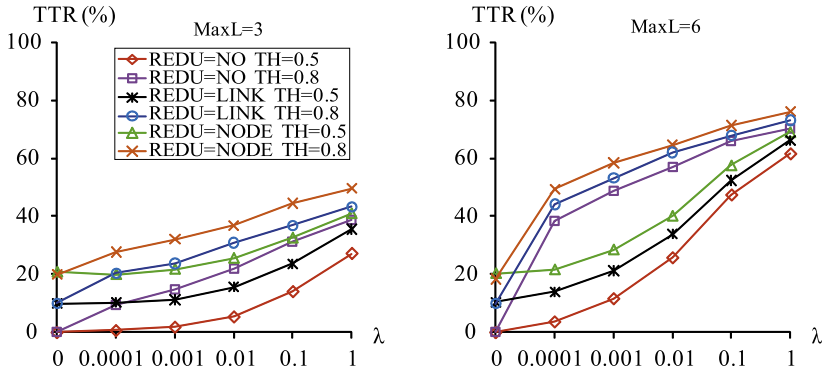

Fig. 7. Trust truncation ratio comparing three approaches.

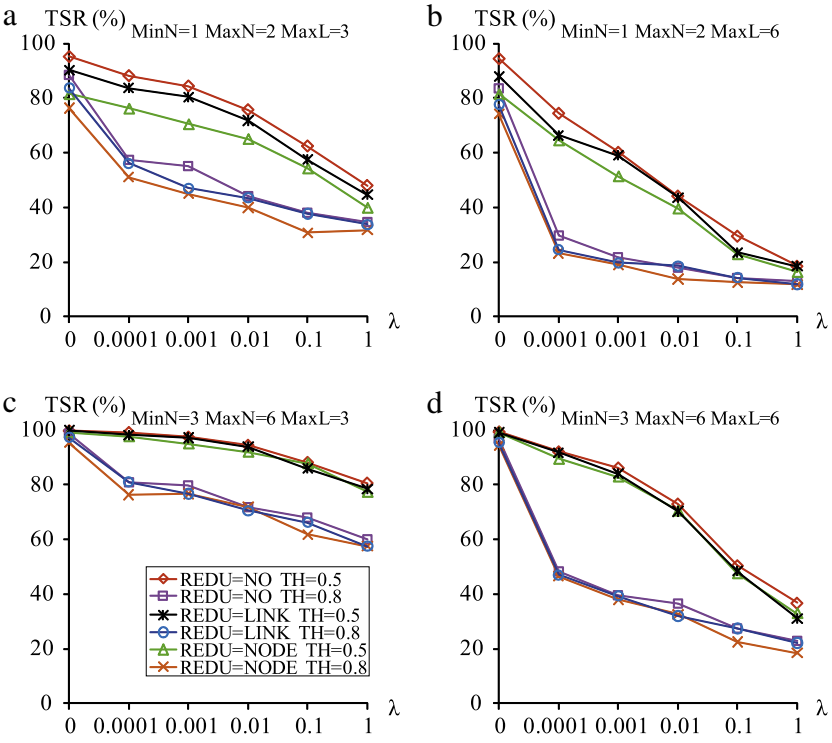

Fig. 8. Trust success ratio comparing three approaches.

promising applications in a variety of self-organizing networks. In such networks, nodes may not know each other, but they need to work together to fulfill users' functionalities and to boost the networking performance. This requires an efficient and effective trust mechanism in such networks and applications. Most existing works apply the same criteria to all users for trust computation, which is counterintuitive concerning the subjective nature and the social phenomenon of trust. We investigated a variety of aspects regarding the personalization of trust, where users can have their own criteria for trust computation based on a trusted graph. However, it remains an open issue to form a trusted graph by finding a set of trusted paths. Chen et al. [21] studied this issue by proposing a heuristic algorithm in P2P networks. Another open issue is how to guarantee that a trusted graph is qualified for trust computation, e.g., the path length of each trusted path found in large networks should be as short as possible. We believe that some existing works on small-world phenomenon can be used for such purposes and we refer to *social distance* [22] and *structural clues* [23] for this research direction. As a conclusion, we have studied the fundamental issues in trust management from node, path, and graph tiers. The proposed *MeTrust* first combines the multi-dimensional evidence and user personalization to broaden the application areas, and it can be easily integrated into different trust management systems for different users. Our extensive experiments show that the proposed scheme can achieve the trust accuracy and the availability of trust computation.

Acknowledgements

This work is supported by the National Natural Science Foundation of China under Grant Nos. 90718034 and 60773013, the Program for Changjiang Scholars and Innovative Research Team in University under Grant No. IRT0661, the Program for New Century Excellent Talents in University under Grant No. NCET-06-0686, and the Hunan Provincial Natural Science Foundation of China for Distinguished Young Scholars under Grant No. 07JJ1010.

References

[1] D. Gefen, Reflections on the dimensions of trust and trustworthiness among online consumers, ACM SIGMIS Database 33 (3) (2002) 38–53.
[2] N. Griffiths, Task delegation using experience-based multi-dimensional trust, in: Proc. of AAMAS 2005, July 2005, pp. 489–496.

[3] L. Mui, M. Mohtashemi, A. Halberstadt, A computational model of trust and reputation, in: Proc. of HICSS 2002, January 2002, pp. 2431–2439.
[4] J. Golbeck, J. Hendler, Inferring binary trust relationships in web-based social networks, ACM Transactions on Internet Technology 6 (4) (2006) 497–529.
[5] T.D. Huynh, A personalized framework for trust assessment, in: Proc. of SAC 2009, March 2009, pp. 1302–1307.
[6] P. Resnick, R. Zeckhauser, E. Friedman, K. Kuwabara, Reputation systems, Communications of the ACM 43 (12) (2000) 45–48.
[7] S.D. Kamvar, M.T. Schlosser, H. GarciaMolina, The EigenTrust algorithm for reputation management in P2P networks, in: Proc. of ACM WWW 2003, May 2003, pp. 640–651.
[8] Y. Wang, A. Nakao, Poisonedwater, an improved approach for accurate reputation ranking in P2P networks, Future Generation Computer Systems (2009) doi:10.1016/j.future.2009.05.001.
[9] R.A. Shaikh, H. Jameel, B.J. d'Auriol, H. Lee, S. Lee, Y.J. Song, Group-based trust management scheme for clustered wireless sensor networks, IEEE Transactions on Parallel and Distributed Systems 20 (11) (2009) 1698–1712.
[10] S. Buchegger, J.-Y. Le Boudec, Performance analysis of the CONFIDANT protocol, in: Proc. of ACM MobiHoc 2002, June 2002, pp. 226–236.
[11] L. Xiong, L. Liu, PeerTrust: supporting reputation-based trust for peer-to-peer electronic communities, IEEE Transactions on Knowledge and Data Engineering 16 (7) (2004) 843–857.
[12] G. Theodorakopoulos, J.S. Baras, On trust models and trust evaluation metrics for ad hoc networks, IEEE Journal on Selected Areas in Communications 24 (2) (2006) 318–328.
[13] A. Jøsang, R. Hayward, S. Pope, Trust network analysis with subjective logic, in: Proc. of ACSC 2006, January 2006, pp. 85–94.
[14] A. Jøsang, T. Bhuiyan, Optimal trust network analysis with subjective logic, in: Proc. of SECURWARE 2008, August 2008, pp. 179–184.
[15] Y. Zuo, W.-C. Hu, T. O'Keefe, Trust computing for social networking, in: Proc. of ITNG 2009, April 2009, pp. 1534–1539.
[16] G. Shafer, A Mathematical Theory of Evidence, Princeton University Press, ISBN: 0-608-02508-9, 1976.
[17] E.P. Klement, R. Mesiar, E. Pap, Triangular Norms, ISBN: 978-0-7923-6416-0, 2000, 406 pages.
[18] M.C. Florea, A.-L. Jousselme, E. Bosse, D. Grenier, Robust combination rules for evidence theory, Information Fusion 10 (2) (2009) 183–197.
[19] Y.L. Sun, W. Yu, Z. Han, K.J.R. Liu, Information theoretic framework of trust modeling and evaluation for ad hoc networks, IEEE Journal on Selected Areas in Communications 24 (2) (2006) 305–317.
[20] D.J. Watts, Six Degrees: the Science of a Connected Age, ISBN: 978-0393325423, 2004, 384 pages.
[21] K. Chen, K. Hwang, G. Chen, Heuristic discovery of role-based trust chains in peer-to-peer networks, IEEE Transactions on Parallel and Distributed Systems 20 (1) (2009) 83–96.
[22] D.J. Watts, P.S. Dodds, M.E.J. Newman, Identity and search in social networks, Science 296 (5571) (2002) 1302–1305.
[23] J. Kleinberg, The small-world phenomenon: an algorithmic perspective, in: Proc. of ACM STOC 2000, May 2000, pp. 163–170.

Guojun Wang received B.Sc. in Geophysics, M.Sc. in Computer Science, and Ph.D. in Computer Science, from Central South University, China. Since 2005, he is a Professor at Central South University. He is the Director of Trusted Computing Institute of the University. He has been an Adjunct Professor at Temple University, USA, a Visiting Scholar at Florida Atlantic University, USA, a Visiting Researcher at the University of Aizu, Japan, and a Research Fellow at the Hong Kong Polytechnic University, Hong Kong. His research interests include trusted computing, mobile computing, pervasive computing, and software engineering. He is an Honor Member of China Computer Federation (CCF) YOCSEF.

Jie Wu is Chair and Professor at the Department of Computer and Information Sciences at Temple University. He is an IEEE Fellow. He is on the editorial board of IEEE Transactions on Mobile Computing. He was a Distinguished Professor at Department of Computer Science and Engineering, Florida Atlantic University. He served as a Program Director at US NSF from 2006 to 2008. He has been on the editorial board of IEEE Transactions on Parallel and Distributed Systems. He has served as a distinguished visitor of the IEEE Computer Society and is the chairman of the IEEE Technical Committee on Distributed Processing (TCDP). His research interests include wireless networks and mobile computing, parallel and distributed systems, and fault-tolerant systems.

Please cite this article in press as: G. Wang, J. Wu, Multi-dimensional evidence-based trust management with multi-trusted paths, Future Generation Computer Systems (2010), doi:10.1016/j.future.2010.04.015