Future Generation Computer Systems 31 (2014) 48–58

Contents lists available at ScienceDirect

Future Generation Computer Systems

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

Generating trusted graphs for trust evaluation in online social networks

Wenjun Jiang [a,b], 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 1 February 2012
Received in revised form
4 June 2012
Accepted 18 June 2012
Available online 26 June 2012

Keywords:
Trusted graph
Trust evaluation
Online social network
Small-world network
Weak tie

ABSTRACT

We propose a novel trust framework to address the issue of "Can Alice trust Bob on a service?" in large online social networks (OSNs). Many models have been proposed for constructing and calculating trust. However, two common shortcomings make them less practical, especially in large OSNs: the information used to construct trust is (1) usually too complicated to get or maintain, that is, it is resource consuming; and (2) usually subjective and changeable, which makes it vulnerable to vicious nodes. With those problems in mind, we focus on generating small trusted graphs for large OSNs, which can be used to make previous trust evaluation algorithms more efficient and practical. We show how to preprocess a social network (PSN) by developing a simple and practical *user-domain-based trusted acquaintance chain discovery* algorithm through using the small-world network characteristics of online social networks and taking advantage of "weak ties". Then, we present how to build a trust network (BTN) and generate a trusted graph (GTG) with the adjustable width breadth-first search algorithms. To validate the effectiveness of our work and to evaluate the quality of the generated trusted graph, we conduct many experiments with the real data set from Epinions.com. Our work is the first that focuses on generating small trusted graphs for large online social networks, and we explore the stable and objective information (such as *domain*) for inferring trust.

1. Introduction

Millions of people are joining online social networks every day, interacting with others who they did not know before. Establishing trust among those indirectly connected users plays a vital role in improving the quality of social network services and enforcing the security for them. The way in which a system discovers, records, and utilizes reputation information to form trust, and uses trust to influence a user's behavior, is referred to as a "reputation and trust-based system" [1]. Reputation and trust systems are seen as "soft security" mechanisms, which use collaborative methods for assessing the behavior of members in the community against the ethical norms, making it possible to identify and sanction those participants who breach the norms, and to recognize and reward members who adhere to the norms [2]. Two common shortcomings make previous trust systems less practical, especially in large OSNs, that is, the information used to construct trust is (1) usually too complicated to get or maintain—it is resource consuming; and (2) usually subjective and changeable, which makes it vulnerable to vicious nodes. With those problems in mind, we focus on generating small trusted graphs for large OSNs, which can be used to make previous trust evaluation algorithms more efficient and practical, as well as resistant to vicious attacks.

1.1. Application scenario and our motivation

Most interactions between two users in online social networks can be broken down into the scenario seen in Fig. 1: Alice is a *service requester*, and Bob is a *service provider*. Bob is the *target* whose trust is to be evaluated along with the *topic* of one of his services, and Alice's question, "Can I trust Bob on this service?"

An effective trust evaluation algorithm is expected to provide a proper answer for Alice. However, most of the existing trust evaluation algorithms are only effective on small-scale networks, which are often represented by several short trusted paths or small trusted graphs. Furthermore, how we can get such small trusted graphs has not been solved in any related literature. Therefore, there is a large gap between large social networks and small trusted graphs [3,4]. Our motivation is to bridge this gap.

In this paper, we mainly focus on Web-based social networks where users can provide user-generated content and construct a list of trusted neighbors, and most importantly, the content can be classified into categories by design. The categories are important, for they will be used to define the users' domain,

* Corresponding author. Tel.: +86 731 88877711; fax: +86 731 88877711.
E-mail addresses: wenjj8a@gmail.com (W. Jiang), csgjwang@gmail.com, csgjwang@mail.csu.edu.cn (G. Wang), jiewu@temple.edu (J. Wu).

0167-739X/$ – see front matter © 2012 Elsevier B.V. All rights reserved.
doi:10.1016/j.future.2012.06.010

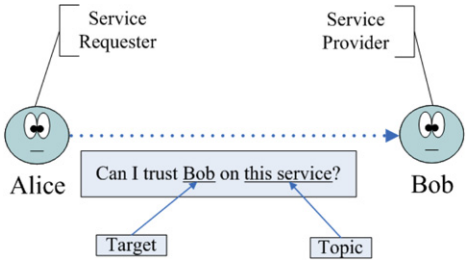

Fig. 1. A scenario of trust evaluation.

which will be further used to compute the social distance between a user and his neighbor. Based on this, we propose the PSN (preprocessing the social network) algorithm. The fundamental theory is Granovetter's [5] famous theory (a highly-influential sociology paper with over 20,000 citations, according to Google Scholar). This theory, known as "The Strength of Weak Ties", discussed the spread of information in social networks. In his theory, it is discovered that weak ties are dramatically valuable, because they are usually the source of new information.

Let us take a look at the scenario in Fig. 1. To make a decision about whether or not to trust Bob, it is natural for Alice to ask her neighbors for suggestions. Next, her neighbors will ask their own neighbors. They will continue to repeat this asking process until they connect with someone who knows Bob. This process is a typical breadth-first search.

It has been widely documented that social networks bear significant traits of small-world networks [6]. Small-world network theory tells us that there exist short path(s) for any two persons (at least most, if not all) in the world. However, in online social networks, a user usually has hundreds of neighbors, which results in the high complexity of the breadth-first search. So, it is very challenging to efficiently locate the short paths hidden within the enormous network.

Some existing research provided a solution that sets a limit to the search length, based on small-world network theory, such as the work in [7,8]; others made use of small-world network models for analysis or simulation. For instance, Watts's small-world network model [6] was used in [9,10]. In this paper, we propose a flexible approach in which the width of the breadth-first search is adjustable. Our key idea is: (1) to select long contacts to reach the target quickly, according to the theory of "weak ties"; and (2) to discover capable neighbors who can give effective suggestions in each step of the breadth-first search based on their topic-related degrees and target-related degrees.

1.2. The concept of trust

We take the natural definition of "trust", that is, Golbeck's [9] definition in the context of a social web where "trust in a person is a commitment to an action based on a belief that the future actions of that person will lead to a good outcome".

We also distinguish "referral trust" from "functional trust", which were first proposed by Jøsang [11]. In our work, "referral trust" represents the ability to recommend a good target, while "functional trust" represents the true ability of a target from his direct neighbor's point of view. For example, Alice needs to have her car serviced, so she asks Bob for his advice about where to find a good car mechanic in town. Bob is thus trusted by Alice to know about a good car mechanic (which is called "referral trust"). Also, Bob trusts Eric to be a good car mechanic because of his direct experience; this kind of trust is called "functional trust".

Another challenge with trust evaluation is taking proper information for inferring trust through trusted paths. Frequently used information, such as reputation, similarity, and explicit ratings, is often subjective, and can be easily changed by users. In this paper, we explore stable and objective information for inferring trust.

1.3. Our contribution

Based on the above analysis, we propose a trust framework called *SWTrust*. In this paper, we do not extend our work into developing integrated trust evaluation models, but rather, we focus on generating trusted graphs from large online social networks that could then be incorporated in the existing models to make them more efficient and practical. Our key ideas and contributions are as follows:

- In order to solve the key challenge of "discovering short trusted paths efficiently", we propose a novel *user-domain-based trusted acquaintance chain discovery* algorithm for preprocessing a large social network (*PSN*), based on the theory of "weak ties".
- We provide both centralized and distributed breadth-first search algorithms for building a trust network, based on the trusted acquaintance chains discovered by the *PSN* process. Moreover, we differentiate between *referral trust* and *functional trust*, and explore the stable and objective information (such as *domain*) for inferring trust, which can weaken the effect of vicious nodes.
- We conduct a lot of experiments with the data set from Epinions.com, and the results show the effectiveness of our work. In addition, we obtain many interesting and useful findings from the experiments.

The remainder of this paper is organized as follows: Section 2 surveys related work in the literature and presents our approach. Section 3 describes the overview of the *SWTrust* framework. Sections 4–6 respectively describe one of the three key steps of the *SWTrust* framework. Section 7 describes the experimental design, shows the numerical results that are obtained through the evaluation of *SWTrust*, and provides their physical interpretations. Finally, Section 8 concludes this paper and suggests future work.

2. Related work

Relationship-oriented networking tries to provide better security, usability, and trust in the system, and allows different users and institutions to build trust relationships within networks similar to those in the real world [12]. Online social networks are one of such networks. Therefore, building trust relationships is a key issue for online social networks.

Wang and Wu proposed FlowTrust to infer trust with network flows [3] and MeTrust for trust management with multi-trusted paths, based on multi-dimensional evidence [4]. In both papers, they presented the strong necessity of generating small trusted graphs.

Mármol and Pérez [13] summarized features of trust and reputation models, in which "gathering behavioral information, scoring and ranking entities, entity selection, transaction, and rewarding and punishing entities" are five components of a complete model. In this paper, we do not extend our work into developing an integrated trust evaluation model, but rather, we focus on generating trusted graphs that could then be incorporated in the existing models, which can be taken as the first three components of a complete model.

2.1. Local and global trust

The advantages and disadvantages of local and global trust metrics are discussed in detail in [7]. The authors pointed out that the local trust metric is more accurate when considering a personal view. In this paper, we mainly go the way of the local

metric, i.e., computing personalized trust value with mainly local information, which is also consistent with small-world network theory.

However, to compute trust for all users, a local trust metric is more complex than a global one. Speeding up the web servers is one of the main architectural challenges of the existing social websites [14]. A more reasonable setting for a local trust metric would be one in which every user runs it from his personal point of view [7]. In this work, we provide both centralized and distributed algorithms. In a distributed setting, each user can select next hops, based on his own knowledge of connected neighbors, using his own computer or mobile device.

2.2. Transitivity of trust

The primary property of trust used in this work is transitivity. That is, if A highly trusts B, and B highly trusts C, then it is with high probability that A will trust C.

Transitive trust inference, usually based on graph theory, is facing the challenge of finding short and trusted paths for two given users. The challenge is even more serious in online social networks, because each user may have dozens or hundreds of neighbors; choosing whom will reach the target is really difficult to decide. The main work of this paper is to address this challenge. Granovetter [5] presented the famous theory, "The Strength of Weak Ties", on the spread of information in social networks. Our work in this paper also takes advantage of weak ties in that we differentiate between local neighbors and long contacts of a user, and we select more long contacts when discovering trusted acquaintance chains.

When measuring trust based on path discovery, the length of a path can become an issue [15]. Golbeck found that it is more accurate to predict trust with a shorter path [9]. Jøsang et al. [16] also pointed out the possibility that trust can be diluted through the propagation process. The longer a chain of trusting a user is, the weaker the predicted trust.

However, Lesani and Montazeri [17] suggested that the information inferred from a long chain of people with high trust values may be much more precise than the information inferred from a short chain of people with low trust values. Cho et al. [18] defined trust availability and path reliability, and identified the optimal length of a trust chain that generates the most accurate trust without revealing risk, which is based on a tradeoff between trust availability and path reliability over trust space.

Kim et al. [15] compared and evaluated how the length of available trusted paths and aggregation methods affect prediction accuracy. They proposed four strategies to predict the level of trust and evaluate the prediction accuracy: the strategy for weighted mean aggregation among shortest paths, min–max aggregation among shortest paths, weighted mean aggregation among all paths, and min–max aggregation among all paths. Among those four, they found that the strategy of weighted mean aggregation among all paths is the best.

We implement the four strategies in [15] to test the effectiveness of our work, and we extend it in three ways: firstly, we generate trusted graphs for large online social networks, making use of its own characteristics based on the theory of "weak ties"; secondly, we evaluate the quality of trusted graphs with the extension of eight strategies in Table 4 for predicting trust, using the data from the real online social network of Epinions.com; furthermore, besides path length and trust evaluation strategies, we test the effects of several other factors that may influence the accuracy.

2.3. Information for trust in online social networks

Caverleea et al. [19] presented the SocialTrust framework for providing a network-wide perspective on the trust of all users in online social networks. In SocialTrust, the authors studied three key factors for trust establishment in online social networks of trust group feedback, distinguishing the user's relationship quality from trust, and tracking user behavior, and describing a principled approach for assessing each component.

Skopik et al. [20] presented a novel approach addressing the need for flexible discovery and the involvement of experts and knowledge workers in distributed and cross organizational collaboration scenarios, in which they focused on the notion of social trust in collaborative networks, and demonstrated the inference of trust depending on captured collaboration data that considers individual trust perceptions.

De Meo et al. [21] proposed a general approach to recommend similar users, resources, and social networks to a user, which operates in a social internetworking context instead of on a single social network, considering both explicit and implicit relationships and taking into account both local and global information.

Kim et al. [22] proposed a trust-prediction framework in rating-based experience sharing social networks, which can work even without a web of trust. Their work measures a degree of trust based on users' expertise and preferences regarding topics (i.e., categories), using users' feedback rating data, which is available and much denser than a web of trust.

While the above ideas are good, the information for trust is difficult to get or maintain. In this paper, we propose a simple and practical method that needs little information for inferring trust. Thanks to the theory of "weak ties", we find *domain*, which is objective and stable for use.

3. Overview of *SWTrust*

3.1. Terms and problem definition

In this paper, we use the following terms.

Trust network: a *trust network* can be formed based on transitive trust, with each link representing the trust relationships between two participants.

Trusted graph: a *trusted graph* is a sub-network of a trust network, starting from a *trustor*, ending at a *trustee*, and connected by a set of trusted paths.

Trustor: a *trustor* is a user who wants to know whether to trust someone else and starts the transitive trust evaluation.

Trustee: a *trustee* is a user who is being considered for interaction and is the end of a transitive trust evaluation.

Domain: a *domain* is a basic unit in which members share some specific interests, such as a sub-category of goods in Epinions.com.

We want to emphasize that *domain* is different from *group* in some other research, in that *domain* is extracted from the architecture of online social websites, while *group* is usually user-defined; correspondingly, the number and attributes of *domain*s are relatively stable, while those of *group*s are usually changeable.

Domain is also different from *attribute*, which is frequently used in the definition of *similarity*, in that *domain* is a broader concept that can reflect the objective behavior of a user, while *attribute* is usually more subjective.

Active domain: a user's *active domain* includes all of the *domain*s in which he has participated; for instance, the place where he wrote reviews of goods and gave ratings to other's reviews, etc.

Topic: a *topic* is the reason why or for what the trustor wants to know the trustee; usually it is a service provided by the trustee.

Let $\mathcal{S}$ be a social network, which can be seen as a large graph. Our work has two sub-tasks. Firstly, given two nodes (*trustor* and *trustee*) of $\mathcal{S}$, find as many short and trusted *trustor–trustee* paths necessary to efficiently construct a subgraph, that is, we generate a trusted graph from a large social network. Secondly, evaluate the quality of a trusted graph. The latter is as challenging as the former because there is no standard test bed for trust models.

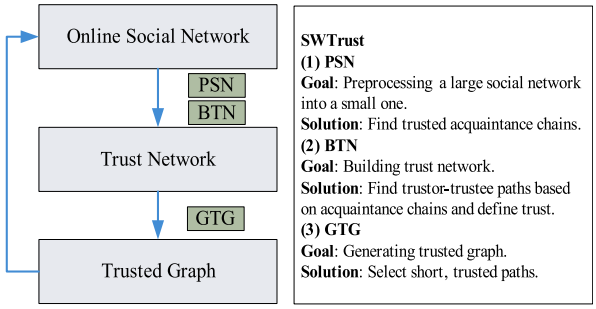

Fig. 2. The architecture of the *SWTrust* framework.

3.2. Architecture

SWTrust has three key steps, namely, preprocessing a social network (PSN), building a trust network (BTN), and generating a trusted graph (GTG) (see Fig. 2). While the architecture is suitable to most online social networks, the following methods are more appropriate for those online social websites in which it is easy to define *domain* from the network architecture, such as Epinions.com.

Currently, we have not dealt with a social network such as Twitter, since it is difficult to define domains/categories for the content in Twitter, which is the key to our proposed user-domain-based trusted acquaintance discovery algorithm. However, after using the data mining technique to preprocess the content in Twitter, which can classify the content into different categories with different topics, our method may be used to predict the trust relationships between users in Twitter.

3.3. Malicious behavior

Reputation and trust systems are seen as "soft security" mechanisms, which use collaborative methods for assessing the behavior of members in the community against the ethical norms, making it possible to identify and sanction those participants who breach the norms, and to recognize and reward members who adhere to the norms [2]. Our work aims to construct a trust evaluation system efficiently. We consider two kinds of malicious behavior: one is a noncollusive setting, that is, a vicious node gives low trust to all of his neighbors; another is a collusive setting, in which a vicious node gives low trust to good nodes and high trust to vicious nodes.

We deal with the attack behaviors in three ways: (1) in each step where we select the next hops, we select from the user's trusted acquaintance list; (2) we take the static and objective information for referring trust along trusted paths, which cannot be changed at will; and (3) we deal with trust conflicts when aggregating the functional trust of direct neighbors.

4. PSN: preprocessing a online social network

Our work in this section is based on the following intuitions:

- When we turn to someone for help, we always choose someone who is related to the thing we are doing, i.e., topic related.
- When we want to know the *target*, we will naturally turn to someone who is related to the *target*, i.e., target related.

4.1. Preparation

As mentioned before, the main goal of our work is to provide an answer to Alice's question of "Can I trust the *target* on this *topic*?" We try to gain further insight and try to determine what Alice will do. Generally, she will turn to some of her friends for suggestions; then comes the next question, "Who will she ask for help?" We find the following common features of those selected friends:

- More or less, they are all related to the *topic*, because only that selection will give her effective suggestions.
- It is better if they are related to the *target*.
- If no friend is related to the *topic* and the *target*, then it is beyond the scope of this paper. We will have nothing else to do other than say "good luck". Alice herself will take the risk of trying. However, it is rare that people will do something completely new if none of their friends know anything about it.

Based on the above insights, we propose the *PSN* method to preprocess the social network before any further steps. We make the following representations for further processing.

- There are a total of N domains in the social network, denoted as $d_1, d_2, \ldots, d_N$ (as mentioned before, a domain is a basic unit); User i's *active domain* is denoted as D_i, which may include several domains. We represent D_i with a binary string, $i_1, i_2, \ldots, i_N$, if D_i includes d_j; then $i_j = 1$, or else $i_j = 0$. For example, when $N = 5$, D_i includes three domains, d_1, d_2, d_5, and then $D_i = 11001$.
- User i has m neighbors, $n_{i1}, \ldots, n_{im}$, and their active domains are denoted as $D_{i1}, \ldots, D_{im}$. Similarly, *topic*'s *active domain* is D_{topic}, and *target*'s *active domain* is D_{target}.

Without loss of generality, we assume that user i has knowledge of the *active domain* of his neighbors, the *topic* and the *target*. We define three operators for computing the relations between user i and his neighbor n_{ij}.

- *Definition* 1: $\diamond$ *operator* counts the number of bits of "1" that have the same position in D_i and D_{ij}, i.e., the number of common domains between user i and his neighbor n_{ij}. For example, $11100 \diamond 11011 = 2$, $11100 \diamond 00011 = 0$.
- *Definition* 2: $\circ$ *operator* counts the number of bits of "1" in D_i, i.e., the number of domains where user i is involved. For example, $\circ 11100 = 3$.
- *Definition* 3: the *social distance* from i to his neighbor n_{ij} is $d(i, ij) = \circ D_{ij} - (D_i \diamond D_{ij}) + 1$. This definition extends the one presented in [23], in which $d(i, ij) = 1$ if i and n_{ij} belong to any common domain. In our work, $d(i, ij) = 1$ only if $D_{ij} \subseteq D_i$. The basic intuition is that: if there is one domain where n_{ij} is involved but user i is not, user i will not know n_{ij}'s behavior in that domain, and consequently cannot give a proper opinion of him when considering that domain. The concept here is similar to the social difference, which we usually use in real life.

Based on the above three new operators, we give more definitions below.

Definition 4: *topic related degree* of user i is $x(i) = D_{topic} \diamond D_i$.

Definition 5: *target related degree* of user i is $y(i) = D_{target} \diamond D_i$.

Definition 6: *the priority* of neighbor j to be selected as the next hop by i is:

$$p(i, j) = \frac{\lambda 1 * x(j) + \lambda 2 * y(j)}{\circ D_{topic} + \circ D_{target}}.$$

Note that $\lambda 1$ and $\lambda 2$ are adjustable parameters in the range of [0, 1], which can be set by each user.

We can see from Definition 6 that j's priority to be selected as the next hop is only related to the topic, the target, and j's topic related degree and target related degree; user i cannot change j's priority at will.

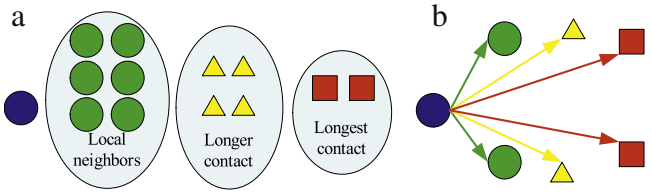

Fig. 3. (a) The three categories of local neighbors, longer, and longest contacts. (b) Selection of next hop neighbors.

4.2. The PSN Process

The *PSN* process is as follows, which we call *user-domain-based trusted acquaintance chain discovery*. Our basic intuition is that long contacts can reach the target more quickly, according to the theory of "weak ties". Thus, we differentiate between local neighbors and long contacts, and we prefer to choose more long contacts when selecting next hops.

- Divide the neighbors of user i into three categories by their *social distance* from i, and one neighbor can be in only one category.
  Category 1: local neighbors. $d(i, ij) = 1$. In this category, $D_{ij} \subseteq D_i$, which indicates that the two users have quite an intimate connection.
  Category 2: longer contact. $d(i, ij) > 1$, and $d(i, ij) < \circ D_{ij}$. In this category, there is more than one (but not all) common basic domain between user i and his neighbor n_{ij}, which indicates that the two users do have a connection, but may not have a very intimate connection. Therefore, n_{ij} is the longer contact of user i.
  Category 3: longest contact. $d(i, ij) > 1$, and $d(i, ij) = \circ D_{ij}$. In this category, there is only one common domain between user i and his neighbor n_{ij}. Thus, n_{ij} is the longest contact of user i.
- In each Category, sort the neighbors with their *priority* in descending order.
- Select next hop neighbors uniformly from the three categories in this way: firstly, choose the one with the highest priority in Category 3; then, choose the one with the highest priority in Category 2; finally, choose the one with the highest priority in Category 1. If it is necessary to choose more nodes, do the same process iteratively with the remaining nodes.

Fig. 3(a) shows the three categories of local neighbors, longer contacts, and longest contacts. Fig. 3(b) shows the priority and category-based selection of next hop neighbors.

Note that neighbors with the same *social distance* may be classified into different categories. The difference is that a neighbor in Category 3 has only one common domain with user i, while in Category 2, it is at least two. The more domains in common, the more similarities between them; correspondingly, the less difference.

5. BTN: building the trust network

To build a trust network from SOURCE to SINK, two things need to be done:

- Find as many short paths for the two given nodes as possible, which is a typical breadth-first search, just as was mentioned before. For each step, we use *PSN* to discover trusted acquaintance chains.
- Add trust information between directly connected nodes.

We provide two methods for the breadth-first search: one is a *centralized* method, which is proper for small-scale online social networks, such as a social network for a university, a company, etc. The other one is a *distributed* method. A distributed approach is more appropriate for large online social networks, which can lighten the burden on the servers.

In the following Algorithms 1–3, let G be the social network after the *PSN* process, let L be the max length of paths, and let c be used to control the path length. Let $L^+(u)$ denote the set of trusted acquaintances of u, which are selected by the *PSN* process and are sorted in descending order by their priorities in each category of longest contacts, longer contacts, and local neighbors. Let R_{source} denote the unvisited nodes.

Algorithm 1 CBFS(G, SOURCE, SINK)

```
1: Input: G. SOURCE, a trustor;  SINK, a trustee.
2: Output: D, a path set from SOURCE to SINK.
3: c ← L − 1. Let SOURCE be the current node.
4: for each neighbor u in R_source of current node do
5:     if u is SINK then
6:         do backtracking to get a path P, add P into D.
7:         Continue to next loop.
8:     else
9:         if c > 0 then
10:            Add all nodes in L+(u) into R_source. c ← c − 1.
11:            Set u as visited.
12:        end if
13:    end if
14: end for
```

Algorithm 2 DBFS(G, SOURCE, SINK)

```
1: Input: G. SOURCE, a trustor;  SINK, a trustee.
2: Output: D, a path set from SOURCE to SINK.
3: c ← L − 1. Let SOURCE be the current node.
4: for each unvisited neighbor u of current node do
5:     if u is SINK then
6:         u sends response backward to his requester, add u to responder; Repeat until a
           response is sent to SOURCE. Reverse responder to get a path P, then add P into D.
7:     else
8:         if c > 0 then
9:             c ← c − 1, u sends request to L+(u).
10:            Set u as visited.
11:        end if
12:    end if
13: end for
```

Algorithm 3 CGTG(D, SOURCE, SINK)

```
1: Input: D. SOURCE, a trustor;  SINK, a trustee.
2: Output: Set D*, a trusted path set from SOURCE to SINK.
3: for each path P in D do
4:     Delete P if the length of P is bigger than L.
5:     Delete P if any RT(i, j) of P is lower than trust threshold.
6: end for
```

5.1. Centralized breadth-first search: CBFS

CBFS (Algorithm 1) is executed in a centralized server, where the network information is stored. Therefore, the server will have to provide both storage and computation. If many users are requesting trust evaluations of someone else, the server will be very busy.

5.2. Distributed breadth-first search: DBFS

In *DBFS* (Algorithm 2), the selection for next hop neighbors is done by the current user. After that, he or she will send a *request* to the selected neighbors to continue this process, and will wait for the *response* back from them. The following representations are prepared for describing the process.

- Let *requester* be the user who wants to know the answer to the question of "how about the trustee's reputation according to the *topic*". Of course, the trustor is the first *requester*.

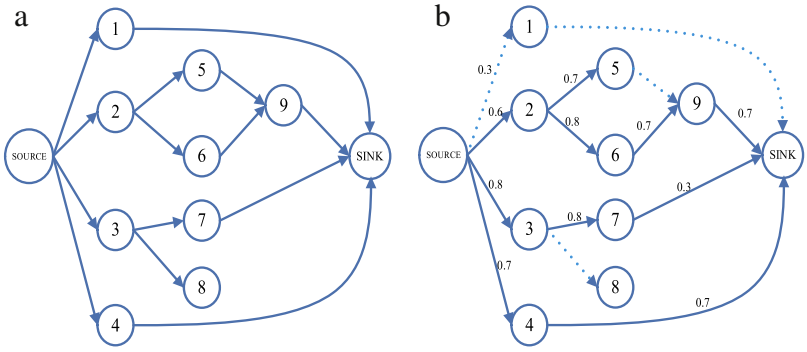

Fig. 4. An example of the BTN and GTG process.

- Let *request* contain the information of *requester, topic, target*, and the value of c.
- Let *response* contain the information of *responder* and the value of c.
- Let *responder* be the sequence of nodes who have sent a *response*. Reverse a *responder* from trustee to trustor will get a trustor–trustee path.

Take Fig. 4(a) for example; after running *CBFS* or *DBFS* on Fig. 4(a), we will get paths: SOURCE-1-SINK, SOURCE-2-6-9-SINK, SOURCE-3-7-SINK, SOURCE-4-SINK. The edge $e(3, 8)$ and node 8 are excluded because they cannot reach SINK. The path SOURCE-2-5-9-SINK is also deleted because when node 5 is being visited, its neighbor 9 has been visited by 6 (who has a higher priority than 5, since $p(2, 6) = 0.8$ and $p(2, 5) = 0.7$), and there is no other neighbor available for 5 to reach the sink.

5.3. Defining trust between directly connected nodes

Our work in this paper differentiates between *referral trust* and *functional trust*, which was presented by Jøsang et al. [11]. Take Scenario 1 for instance; the *referral trust* from Alice to her neighbor (say, B) is the degree to which Alice believes that B can make an appropriate evaluation of Bob, while functional trust is the realistic assessment of Bob by his direct neighbors.

By applying *referral trust* to all of the edges except for the last hop, and *functional trust* to the last hop, we can get the trust network.

However, mining trust information for different social networks is a challenging task that involves data mining and other techniques, which is beyond the scope of this paper.

5.3.1. Referral trust

We define *referral trust* as the *priority* to be selected as the next hop, which is in the range of $[0, 1]$.

Definition 7: *referral trust from i to j* is equal to the priority of j to be selected as the next hop by i:

$$RT(i, j) = p(i, j).$$

The *referral trust*, as it is defined in this paper, only uses the information of *domain*, which is stable and objective, and cannot be changed at will.

5.3.2. Functional trust

Since the main work of this paper is to discover short paths, we only give a suggestion for defining functional trust. We can consider several kinds of factors, such as *social relations, explicit ratings, reputation*, and *similarity*.

- The first factor, *social relations*, has different meanings in different environments. It can be a recommendation from an expert, and a user who has such a recommendation will be easily accepted in the network, as well as in real life. It can also be the role in an organization, for instance, the leader of a country is always to be taken as trustful. It can also be the degree of importance for connectivity, such as the hub nodes.
- The second factor, *ratings*, involves subjective evaluations from a user to others. Examples include ratings used in eBay or Taobao (the most popular e-business website in China).
- The third factor, *reputation*, is a reflection of past behavior.
- The fourth factor, *similarity*, reflects commonalities from one user to another. For example, suppose each user has three attributes, $a1, a2, a3$, and two users have the same value for $a1, a2$, then their *similarity* is $2/3$.

6. GTG: generating the trusted graph

The goal of the *GTG* process is to select short, trusted paths from the trust network. Since the trust network is quite small for processing, we can do this in a centralized way.

6.1. CGTG: centralized GTG

CGTG (Algorithm 3) can be used to generate the trusted graph in a centralized server. Let D represent the resulting path set from the *BTN* process, let $RT(i, j)$ represent the *referral trust* from node i to node j, and let $e(i, j)$ be an edge of a path.

Take Fig. 4(b) as an example. Suppose that we get the trust values as labeled on the edges in Fig. 4(b), and the trust threshold $th = 0.5$. After the *GTG* process, the path SOURCE-1-SINK will be excluded, since the trust value from SOURCE to node 1 is lower than th. Up until now, the trusted graph from SOURCE to SINK is generated.

6.2. Trust conflict of trustee

In a trusted graph, a trustee may have several direct neighbors who may have different opinions of him. Thus, we define the *trust conflict* of a trustee to reflect the inconsistency. We calculate the number of his neighbors who have low trust of him and the number of those who have high trust; then, the quotient is defined as *trust conflict*. This definition can avoid an overly optimistic evaluation.

Definition 9: *trust conflict of trustee*:

$$TC(trustee) = \frac{|n_i : T(n_i, trustee) < th|}{|n_j : T(n_j, trustee) \geq th|}$$
$$(th \text{ is the threshold of } trust).$$

We set the threshold of *trust conflict* as $\beta = 1$. If $TC(trustee) \geq \beta$, which means that more than half (including half) of the neighbors think lowly of the trustee, then the trustor would likely not trust him. Therefore, only when $TC(trustee) < \beta$ and $T(trustor, trustee) > th$ can the trustee be seen as trustworthy.

Take Fig. 4(b) for example; $TC(SINK) = 1/2 = 0.5$, and $T(SOURCE, SINK) > 0.5$, SINK will be seen as trustworthy. We emphasize this concept because inconsistencies are always present in real life.

7. Experiments

7.1. Experimental design

In order to test the performance of our work, we use a standard evaluation technique in machine learning: leave one out. If there is an edge between two nodes (say SOURCE and SINK), that edge is masked and trust is calculated through a trusted graph from SOURCE to SINK; then we compare the calculated value with the masked value.

We mainly consider two metrics: *connection coverage* and *trust accuracy*. *Connection coverage* refers to the ability of the algorithms to provide a prediction. In this case, we compute the percentage of trust relationships that are predictable, that is, we can find at least one path that is shorter than the length threshold between two users. *Trust accuracy* represents the ability of predicting that a user will be trusted or not.

We design two separate groups of experiments for *connection coverage* and *trust accuracy*.

- To test the effects of the *PSN* process on the *connection coverage*, we record the number (denoted as n) of edges that have short paths between the pair of nodes; then, *connection coverage* $=$ n/*total edges*. We carried out four sets of experiments (Table 1) and compared the results.
- Based on the results of the coverage experiments, we further analyze the short paths because we want to know how many next hops are local neighbors, longer contacts, and longest contacts.
- To test the *trust accuracy*, we redefined the accuracy metrics in [8], as shown in Table 2, which includes absolute error, precision, recall, and FScore. We use the FScore metric to measure the accuracy using recall and precision jointly, since there is a trade off between precision and recall.
  In this paper, we use *Trust State* to represent when we only consider whether to trust or not.
  For accuracy experiments, we implement all eight common strategies in Table 4, and we test the separate influence of each possible factor. We also present a new SWTrust* algorithm for comparison, which uses the idea behind the TidalTrust algorithm [9]. In the original TidalTrust algorithm, it first searches shortest paths from a source user to a target user. After that, it backtracks from the target user, level by level, to the source user through previously searched strongest (the path that has the largest trust weight, i.e., most reliable) shortest paths.

7.1.1. Data set and preprocess

We use the data set from Epinions.com [7], which was given directly by Epinions staff to Paolo Massa. Yuan et al. have experimented with, and verified, the small-worldness of this data set [24]. The data set contains 132,000 users who issued 841,372 statements (717,667 trusts and 123,705 distrusts), 1,560,144 articles, and 13,668,319 article ratings. Users and Items are represented by anonymized numerical identifiers.

Epinions is an online community website where users can write reviews about products and services, and can also rate other users' reviews on a numerical scale. A review in this context can be seen as content or service, and a review-writer is a service-provider who wants to share his/her product experiences with community members. The ratings for reviews are given by review-raters who have read the reviews and have evaluated the degree of helpfulness of the reviews. Then, review-raters can be seen as service-users.

Table 1
The four experiments for connection coverage.

Note	Meaning
$k = all$	With *PSN* and selecting all neighbors
$k = 18$	With *PSN* and selecting at most 18 next hops
$k = 9$	With *PSN* and selecting at most 9 next hops
$k = 3$	With *PSN* and selecting at most 3 next hops

Table 2
The accuracy metrics.

Metrics	Computing equation
Absolute error	$\lvert trust\ calculated - actual\ trust \rvert$
Precision	$\frac{At \cap Bt}{Bt}$
Recall	$\frac{At \cap Bt}{At}$
FScore	$\frac{2 \times recall \times precision}{recall + precision}$

$At = \lvert e(i,j) : i\ trusts\ j\ directly \rvert$.
$Bt = \lvert e(i,j) : i\ trusts\ j\ by\ trust\ calculated\ through\ algorithm \rvert$.

The data set is stored in three files. The first file is Trust/Distrust information, in which MY_ID stores the ID of the member who is making the trust/distrust statement, OTHER_ID is the ID of the member being trusted/distrusted, VALUE is either 1 for trust and -1 for distrust, and CREATION is the date on which the trust decision was made.

The second file is Article Author information, in which CONTENT_ID is the object ID of the article, AUTHOR_ID is the ID of the user who wrote the article, and SUBJECT_ID is the ID of the subject that the article is supposed to be about.

The third file is Article Ratings information, in which OBJECT_ID is the object that is being rated, and it is the same as Content_ID in the second file. MEMBER_ID stores the ID of the member who is rating the object. Some other columns, that have little impact on our experiments, are not mentioned here.

We can easily get a social network of trust relationships with the first file, however, *domain* information is difficult to get. Since Epinions.com is organized by goods, we make an assumption that goods with continuous SUBJECT_IDs are in the same domain.

Then, we process the data set to get *domain* information for each user: (1) sort the second file, with SUBJECT_IDs in ascending order. There is a total of 1,560,144 rows, and we choose the first 50,000 (we choose a smaller data set). (2) There are a total of 3,142 un-duplicated SUBJECT_IDs after the filter operation. We suppose that these SUBJECT_IDs are involved with 6 domains; then, more than 500 SUBJECT_IDs are taken into a domain. In this way, each SUBJECT_ID is related with a domain. (3) Return to the second file, where there are a total of 20,075 un-duplicated Author_IDs after the filter operation. We select the first 5,000. Then, we make a program to collect domain information of all of these 5,000 Author_IDs. (4) Finally, come to the first file. We first delete the rows which show distrust (distrust is beyond the scope of this paper). Then, we make another program to filter out the User_IDs that are not included in the 5,000 Author_IDs of Step (3). 51,929 rows remained after delete self cycle (My_ID is the same as Other_ID), and 51,888 rows and 3,168 nodes remain, which are the edges and nodes of our experiments.

7.2. Evaluating the efficiency of generating a trusted graph

7.2.1. The connection coverage

Parameter settings: $\lambda 1 = \lambda 2 = 1$, *max length* varies in the range of [2, 12]. Fig. 5(a) shows the coverage of different numbers of next hops. Fig. 5(b) shows the average computing time of finding short paths for each pair of nodes. As the number of next hops is reduced, the coverage decreases a little, but the computing time decreased dramatically (from 3.5547 to 0.7488 s on average), which shows the effectiveness of the *PSN* algorithm. Moreover, the four sets of

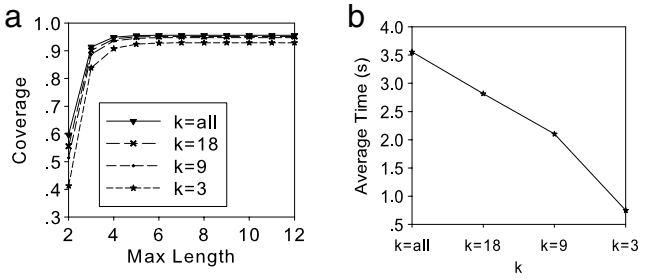

Fig. 5. The effect of the PSN algorithm.

Table 3
The proportion of three categories.

	Local neighbors (%)	Longer contacts (%)	Longest contacts (%)
$k = all$	32.16	27.51	40.33
$k = 9$	31.41	27.45	41.14

experiments reach the max coverage when *max length* = 8, 7, 9, and 10, respectively, which indicates that, if one node can reach the other, the path between them will not be too long.

Furthermore, when $k = 9$, there is a total of 3,525,586 paths for 58,888 pairs of nodes. Thus, there are more than 59 short paths for each pair of nodes on average; most nodes have multiple paths to reach all other nodes. This finding verifies the small-world network theory, in which there exist short paths for any two persons in the world.

7.2.2. The proportion of local neighbors, longer contacts, and longest contacts

Without loss of generality, we use the two path sets of $k = all$ and $k = 9$, with $L = 5$. For each path $(n_1, \ldots, n_i, n_{i+1}, \ldots, n_k)$, we compute the relation of n_i and n_{i+1} to decide if n_{i+1} is the local neighbor, longer contact, or longest contact of n_i. Then, we summarize the total number and compute the proportion.

Table 3 shows the proportion of the three categories of next hops. When $k = all$, long contacts (longer contacts and longest contacts) make up the major proportion of 67.84%; while for $k = 9$, the proportion is 68.59%. The results show that short paths contain the majority of nodes that are long contacts, which verifies our intuition that "long contacts can reach the target quickly" and validates "the strength of weak ties".

7.3. Evaluating the quality of the trusted graph

Since the role of a trusted graph is to predict trust, we compare the performance of several methods for predicting trust based on our generated trusted graph. Furthermore, to be objective, we use the common strategies to evaluate trust instead of giving a new one. If it performs well, we can say it is a graph of high quality.

We focus on three aspects for experiments: (1) whether a generated trusted graph is of high enough quality to predict trust, (2) which factors can influence prediction accuracy, and (3) the robustness of our work.

Common strategies to evaluate trust. There are three factors—propagation method, aggregation method, and length of paths—for common strategies. Each factor has two choices; therefore, there is a total of eight strategies. We implement all of the eight strategies (Table 4) and compare their performances. There are two basic operations for evaluating trust based on trusted paths:

- Trust *propagation* from SOURCE to the direct neighbor of SINK through a trusted acquaintance chain; since trust usually decays along a path, the propagation operator should reflect this point. Two commonly used propagation functions are *Min* and *Multiply*, which entail taking the minimal *referral trust* or taking the products of all the *referral trust*.

Table 4
The common eight strategies for evaluating trust.

ID	Strategy	Propagation	Aggregation	Paths
1	Min–Max	Min	Max	All
2	Multi-Max	Multi	Max	All
3	Min-WAve	Min	Weighted average	All
4	Multi-WAve	Multi	Weighted average	All
5	SMin–Max	Min	Max	All shortest
6	SMulti-Max	Multi	Max	All shortest
7	SMin-WAve	Min	Weighted average	All shortest
8	SMulti-WAve	Multi	Weighted average	All shortest

Table 5
The experimental parameters for accuracy.

Parameter	Description	Default
μ	Mean of normal distribution	–
σ	Standard deviation	0.25
vrate	% of vicious nodes	25%
grate	% of good nodes ($\mu = 0.75$)	70%
brate	% of bad nodes ($\mu = 0.25$)	30%
th	Trust threshold	0.5

Take Fig. 4(b) and the trusted path SOURCE-2-6-9 for instance; the *propagation* operation with *Min* will be $\min(0.6, 0.8, 0.7) = 0.6$, while with *Multiply*, it will be calculated as $0.6 * 0.8 * 0.7 = 0.336$. Both of the two functions satisfy the rule of trust decay along a path.

- Trust *aggregation* among direct neighbors. Two commonly used aggregation functions are *Max* and *Weighted Average*, which entail taking the direct trust of the neighbor in the trusted acquaintance chain with the max *referral trust* and the weighted average value among all direct neighbors.

Determining trust value. To assign trust value to nodes, we introduce Richardson's technique [25] in the experiments, which uses the concept of the quality of users assigning a trust value to each node, while making a little modification (the same method as in [8]). Each user has a quality measurement $q_i \in [0, 1]$. For the experiments in this paper, the quality of a user is chosen from a normal distribution. For any pair of users, i and j, the trust rating from node i to node j, denoted as $T(i, j)$, is uniformly chosen from $[\max(q_j - \delta_{ij}, 0), \min(q_j + \delta_{ij}, 1)]$. In addition, $\delta_{ij} = (1 - q_i)/2$ is a noise parameter that determines how accurate users are at estimating the quality of the user that they are trusting.

Malicious behavior. In the experiments, we consider two kinds of malicious behavior: one is the noncollusive setting, that is, a vicious node gives low trust to all of his neighbors; another is the collusive setting, in which a vicious node gives low trust to good nodes and high trust to vicious nodes.

Default experimental parameters are set as in Table 5.

Without loss of generality, in all of the experiments for *trust accuracy*, we use the path set of $k = 9$. We present the results for the influence factor of trust prediction and the robustness of our work, all of which can reflect the quality of the generated trusted graphs. First of all, we test the methods of defining trust, since the trust value assigned to each edge will affect all the experiments.

7.3.1. Test the influence factors

The methods of defining trust. We design two settings to assign trust value for a trusted graph: one is homogeneous, and the other is heterogeneous. In a homogeneous setting, trust values of all edges are given by Richardson's technique [25], indiscriminate of *referral trust* and *functional trust*. While in a heterogeneous setting, only the last edge, which ends with SINK, is given a *functional trust* value by Richardson's technique [25], and other edges are given a *referral trust* value by the way we define in this paper.

Parameter settings: $L = 6$, $brate = 0.3$, $th = 0.5$. Here, we present the results of two sets of experiments; one is $vrate = 0$,

Table 6
Comparison between heterogeneous and homogeneous settings ($vrate = 0$).

Setting	Method	Mean error	Precision	Recall	FScore
Heterogeneous	Min–Max	0.1346	0.8855	0.8946	0.8900
	Multi-Max	0.1351	0.8917	0.8982	0.8949
	Min-WAve	0.1018	0.9254	0.9294	0.9274
	Multi-WAve	0.1013	0.9239	0.9296	0.9268
	SMin–Max	0.1365	0.8913	0.8894	0.8903
	SMulti-Max	0.1374	0.8909	0.9001	0.8955
	SMin-WAve	0.1203	0.8989	0.9162	0.9075
	SMulti-WAve	0.1205	0.9025	0.9121	0.9073
Homogeneous	Min–Max	0.1261	0.9051	0.9105	0.9078
	Multi-Max	0.1308	0.9024	0.9134	0.9079
	Min-WAve	0.1145	0.9092	0.9182	0.9137
	Multi-WAve	0.1156	0.9118	0.9142	0.9130
	SMin–Max	0.1263	0.9013	0.9164	0.9088
	SMulti-Max	0.1279	0.8991	0.9131	0.9060
	SMin-WAve	0.1220	0.9006	0.9167	0.9086
	SMulti-WAve	0.1224	0.8976	0.9140	0.9057

Table 7
Comparison between heterogeneous and homogeneous settings ($vrate = 0.25$).

Setting	Method	Mean error	Precision	Recall	FScore
Heterogeneous	Min–Max	0.1974	0.7996	0.7933	0.7964
	Multi-Max	0.1969	0.7967	0.7874	0.7920
	Min-WAve	0.1500	0.8967	0.8189	0.8560
	Multi-WAve	0.1496	0.8925	0.8166	0.8529
	SMin–Max	0.1982	0.7996	0.7933	0.7964
	SMulti-Max	0.1970	0.7975	0.7899	0.7937
	SMin-WAve	0.1770	0.8381	0.7996	0.8184
	SMulti-WAve	0.1752	0.8356	0.8023	0.8186
Homogeneous	Min–Max	0.1989	0.7482	0.8357	0.7896
	Multi-Max	0.2010	0.7421	0.8388	0.7875
	Min-WAve	0.1730	0.7940	0.8515	0.8217
	Multi-WAve	0.1777	0.7862	0.8416	0.8130
	SMin–Max	0.1995	0.7532	0.8362	0.7925
	SMulti-Max	0.1964	0.7635	0.8373	0.7987
	SMin-WAve	0.1825	0.7709	0.8371	0.8027
	SMulti-WAve	0.1870	0.7672	0.8354	0.7998

and the other is $vrate = 0.25$, and vicious nodes are collusive (we conduct experiments in all possible settings, since those results have a similar pattern when considering the methods of defining trust; here, we just present one of them).

Tables 6 and 7 show the comparison when $vrate = 0$ and $vrate = 0.25$. We can see that the accuracy is high with the min FScore of 0.7875. It indicates that the generated trusted graphs are of high quality when it comes to predicting trust.

We also observed some interesting details: (1) in both settings, $vrate = 0$ and $vrate = 0.25$, the accuracy of using the *Multi* propagation function has little difference with that of using the *Min* function. On the contrary, the aggregation method has a more significant effect. In both settings, the accuracy of using the *WAve* function is higher than that of using the *Max* function. The gap is larger in the heterogeneous setting. This phenomenon exists in all of the accuracy experiments, which indicates that opinions from multiple paths are better than that from a single path, so as to avoid being subjective and one-sided. (2) The accuracy is higher in the heterogeneous setting in most cases. Take the *min* function for instance; the percentage of a higher FScore in the heterogeneous setting compared to the homogeneous setting is: 4.17% when using the *WAve* function, 0.86% when using the *Max* function in the setting of $vrate = 0.25$ (see Table 7), and 1.5% when using the *WAve* function in the setting of $vrate = 0$. The results validate our intuition that distinguishing between *referral trust* and *functional trust*, as well as exploring the stable and objective information for inferring trust, can weaken the effect of vicious nodes.

In the following experiments, we use the heterogeneous setting to determine trust value, that is, the direct neighbors of SINK give their *functional trust* to SINK, while others give their *referral trust* to neighbors. In all of the accuracy experiments, we found that the two propagation methods of *Min* and *Multiple* make little difference to the accuracy; therefore, in the following experiments, we only present the results of the *Min* propagation.

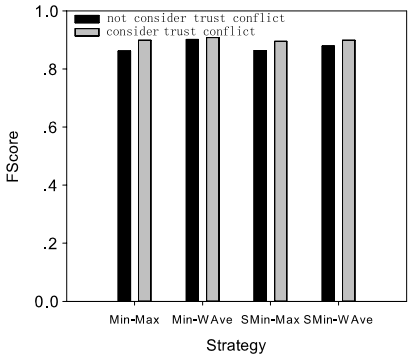

Fig. 6. The effect of trust conflict.

Trust conflict. Parameter settings: $L = 6, brate = 0.3, th = 0.5$. Fig. 6 shows the effect of considering and not considering *trust conflict*. We can see that: when using the *Max* function to do aggregation, the accuracy increases more when considering *trust conflict*. It indicates that *trust conflict* can weaken the negative effect of the one-sidedness of using the *Max* function.

Max length. Parameter settings: L varies in the range of [2, 8], $th = 0.5, brate = 0.3$. Fig. 7(a) and (b) show the comparison of FScore and mean error with different max lengths. We can observe

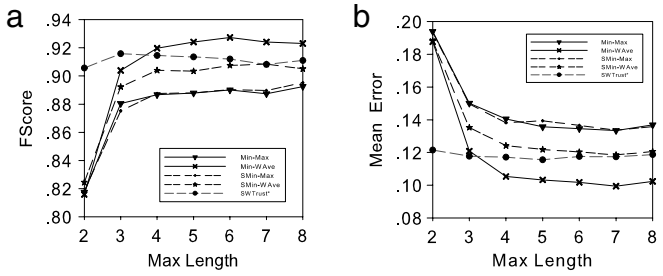

Fig. 7. The effect of max length.

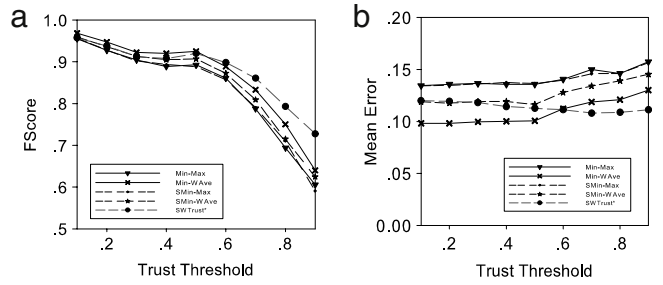

Fig. 8. The effect of trust threshold.

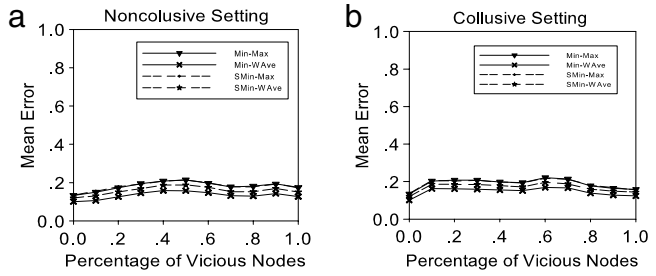

Fig. 9. Mean error with respect to vicious nodes.

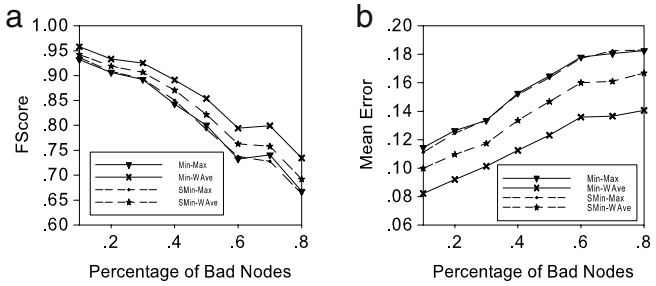

Fig. 10. The effect of the quality of nodes.

that the accuracy is increased along with L, until $L = 6$. When $L = 2$, the accuracy is the lowest, while when $L > 6$, the accuracy goes down a little. We analyzed the reason and found that there are few paths when L is too small; searching with a larger length will find more paths, which means more evidence. However, if $L > 6$, too many paths with low trust will be taken in and will decrease the accuracy. It validates again that a trusted graph with multiple paths is better than a single path when predicting trust; In addition, trusted paths should be neither too long nor too short.

We also found that the accuracy increases much quicker when L is smaller. The reason is that we can find less new paths as L gets larger. Since the average path length of the data set is 3.96, according to the work of [24]; as to our experiments of $k = 9$, the average length is 4.0015.

Trust threshold. Parameter settings: $L = 6, brate = 0.3, th$ varies in the range of [0.1, 0.9]. Fig. 8(a) and (b) show the FScore and mean error with respect to the trust threshold. Quite surprisingly, when th varies from 0.1 to 0.9, the accuracy decreases, especially when $th \geq 0.6$—FScore decreases sharply and mean error increases sharply. It indicates that *trust threshold* should not be set to too large a value, since a larger *trust threshold* will make more paths untrustful and will be deleted. Opinions from less paths can easily be one-sided; furthermore, *referral trust*, as defined in this paper, is related to a neighbor's topic-related degree and target-related degree. It is quite possible that all neighbors are not very familiar with the topic and the target; however, we can still select the most related one to give relatively effectual suggestions.

7.3.2. Test the robustness

Malicious behavior. Parameter settings: $L = 6, brate = 0.3, th = 0.5, vrate$ varies from 0.1 to 1.0, with an increment of 0.1, in collusive and noncollusive settings. Fig. 9(a) and (b) show the mean error with respect to different percentages of vicious nodes. In both noncollusive and collusive settings, the mean of trust computation error increases slowly with the percentages of vicious nodes, even when all of the nodes become vicious. This indicates that the generated trusted graph cannot eliminate, but can weaken, the effect of vicious nodes. Interestingly enough, the mean error reaches its highest point when 50% of nodes are vicious, which indicates the behavior difference is largest when half of the nodes are vicious and the other half are good.

The quality of nodes. We test the effect of the quality of nodes, by varying the proportion of good nodes and bad nodes. Parameter settings: $L = 6, th = 0.5, brate$ varies in [0.1, 0.8]. A user's quality is chosen from two normal distributions of $\mu = 0.25$ (which represents bad nodes) and $\mu = 0.75$ (which represents good nodes), according to Richardson's technique [25]. Fig. 10(a) and (b) show the FScore and mean error with respect to different proportions of bad nodes and good nodes. From the results, we can see that the accuracy is relatively high: even when 70% of nodes are bad nodes, the FScore is larger than 0.75. This finding again validates that the generated trusted graphs are of high enough quality to predict trust.

7.3.3. Comparative study

As mentioned before, to be fair and objective, we implemented the common strategies to evaluate trust based on our generated trust graphs, instead of giving a new integrated and comprehensive predicting approach. If it performs well, we can say that the generated trust graph is of high quality. However, to ensure the true effects of our work, we implemented a new trust prediction strategy denoted as SWTrust*, which takes the weighted average of all shortest and strongest paths, based on our generated trust graph. Figs. 7 and 8 show the trust prediction accuracy with respect to the max length and trust threshold. The results indicate that SWTrust* has a better and more stable performance compared to the common strategies, for its FScore keeps a high and stable level while the mean error keeps a low and stable level during the whole range of max length and trust threshold. SWTrust* utilizes the idea of TidalTrust [9]. In the original TidalTrust algorithm, each node has its trust measurement of the target, which means it needs more information. However, in most trust chain-based scenarios, it is difficult for indirect neighbors to provide their trust to the target. Therefore, it is difficult to make a fair comparison with the original TidalTrust algorithm.

7.4. Summary of experiments

To validate the effectiveness of our work, we conduct two groups of experiments with the data from Epinions.com. The experiments for connection coverage show the efficiency of *SWTrust* and verify the small-world network theory in that

the coverage is more than 92.8% when *max length* = 6; the experiments also verify the effect of weak ties in that the proportion of long contacts is 67.84% when selecting all neighbors for the breadth-first search. The experiments for accuracy show that the generated trusted graphs perform well when predicting trust. In addition, we get many useful findings: (1) from the experiments that compare heterogeneous and homogeneous methods to determine trust value, we observed that using stable and objective information to infer trust can weaken the effect of vicious nodes, since the information cannot be changed at will. (2) From the experiments that compare different max lengths of trust paths and aggregation methods of *Max* and *WAve*, we reach the conclusion that an opinion from multiple paths is better than that from a single path, to avoid being subjective and one-sided. (3) When comparing the accuracy of experiments with and without the consideration of trust conflict, we observe that there do exist controversial users. Introducing the new factor of *trust conflict* can increase accuracy, especially when using the *Max* function to do aggregation, since *trust conflict* can weaken the negative effect of one-sidedness.

8. Conclusion and future work

In this paper, we propose the *SWTrust* framework to discover short trusted paths based on trusted acquaintance chains, and we generate trusted graphs for online social networks. Previous trust evaluation algorithms can become more efficient and practical if they apply *SWTrust* to generate trusted graphs. We focus on generating trusted graphs and explore stable and objective information in order to select capable neighbors in each step of the breadth-first search. Experiments with a data set from a real online social network validate the effectiveness of our work. Exploring stable and objective information (such as *domain*) for different kinds of online social networks, as well as designing a new integrated and comprehensive model for predicting trust, are our main future work.

Acknowledgments

This work is supported in part by the National Natural Science Foundation of China under grant numbers 61073037 and 61103035, and the Ministry of Education Fund for Doctoral Disciplines in Higher Education under grant number 20110162110043.

References

[1] A. Srinivasan, J. Teitelbaum, H. Liang, J. Wu, M. Cardei, Reputation and trust-based systems for ad hoc and sensor networks, in: A. Boukerche (Ed.), Algorithms and Protocols for Wireless, Mobile Ad Hoc Networks, Wiley, ISBN: 978-0-470-38358-2, 2008.
[2] A. Jøsang, R. Ismail, C. Boyd, A survey of trust and reputation systems for online service provision, Decision Support Systems 43 (2) (2007) 618–644.
[3] G. Wang, J. Wu, FlowTrust: trust inference with network flows, Frontiers of Computer Science in China 5 (2) (2011) 181–194.
[4] G. Wang, J. Wu, Multi-dimensional evidence-based trust management with multi-trusted paths, Future Generation Computer Systems 27 (5) (2011) 529–538. (Elsevier).
[5] M. Granovetter, The strength of weak ties: a network theory revisited, Sociological Theory 1 (1983) 201–233.
[6] D.J. Watts, Small Worlds: The Dynamics of Networks Between Order and Randomness, Princeton University Press, Princeton, NJ, 1999.
[7] P. Massa, P. Avesani, Trust metrics on controversial users: balancing between tyranny of the majority and echo chambers, International Journal on Semantic Web and Information Systems 3 (2007) 39–64.
[8] S. Shekarpour, S.D. Katebi, Modeling and evaluation of trust with an extension in semantic web, Web Semantics: Science, Services and Agents on the World Wide Web 8 (1) (2010) 26–36.
[9] J. Golbeck, Computing and applying trust in web-based social networks, Ph.D. Thesis, University of Maryland, 2005.
[10] 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.
[11] A. Jøsang, R. Hayward, S. Pope, Trust network analysis with subjective logic, in: Proceedings of the 29th Australasian Computer Science Conference, ACSC2006, January 2006, pp. 85–94.
[12] S. Paul, J. Pan, R. Jain, Architectures for the future networks and the next generation Internet: a survey, Computer Communications 34 (1) (2011) 2–42.
[13] F. Gómez Mármol, G. Martínez Pérez, Towards pre-standardization of trust and reputation models for distributed and heterogeneous systems, Computer Standards & Interfaces 32 (4) (2010) 185–196.
[14] K. Won, J. Ok-Ran, L. Sang-Won, On social web sites, Information Systems 35 (2) (2010) 215–236.
[15] Y.A. Kim, H.S. Song, Strategies for predicting local trust based on trust propagation in social networks, Knowledge-Based Systems 24 (8) (2011) 1360–1371.
[16] A. Jøsang, E. Gray, M. Kinateder, Simplification and analysis of transitive trust networks, Web Intelligence and Agent Systems 4 (2) (2006) 139–161.
[17] M. Lesani, N. Montazeri, Fuzzy trust aggregation and personalized trust inference in virtual social networks, Computational Intelligence 25 (2) (2009) 51–83.
[18] J.-H. Cho, A. Swami, I.-R. Chen, Modeling and analysis of trust management with trust chain optimization in mobile ad hoc networks, Journal of Network and Computer Applications 35 (3) (2012) 1001–1012. http://dx.doi.org/10.1016/j.jnca.2011.03.016.
[19] J. Caverleea, L. Liu, S. Web, The socialtrust framework for trusted social information management: architecture and algorithms, Information Sciences 180 (1) (2010) 95–112.
[20] F. Skopik, D. Schall, S. Dustdar, Modeling and mining of dynamic trust in complex service-oriented systems, Information Systems 35 (7) (2010) 735–757.
[21] P. De Meo, A. Nocera, G. Terracina, D. Ursi, Recommendation of similar users, resources and social networks in a Social Internetworking Scenario, Information Sciences 181 (2011) 1285–1305.
[22] Y.A. Kim, R. Phalak, A trust prediction framework in rating-based experience sharing social networks without a web of trust, Information Sciences 191 (2012) 128–145.
[23] D.J. Watts, P.S. Dodds, M.E.J. Newman, Identity and search in social networks, Science 296 (5571) (2002) 1302–1305.
[24] W.W. Yuan, D.H. Guan, Y.-K. Lee, Improved trust-aware recommender system using small-worldness of trust networks, Knowledge-Based Systems 23 (2010) 232–238.
[25] M. Richardson, R. Agrawal, P. Domingos, Trust management for the semantic web, in: Proceedings of International Semantic Web Conference, ISWC 2003, vol. 2870, 2003, pp. 351–368.

Wenjun Jiang received the Bachelor's degree in Computer Science from Hunan University, PR China, in 2004; and the Master's degree in Computer Software and Theory from Huazhong University of Science and Technology, PR China, in 2007. She has been a Ph.D. candidate at Central South University since September 2009. Her research interests include trust evaluation models and algorithms in online social networks.

Guojun Wang received the B.Sc. degree in Geophysics, the M.Sc. degree in Computer Science, and the Ph.D. degree in Computer Science, from Central South University (CSU), China. He is currently Chairman and Professor of the Department of Computer Science at CSU, and Director of Trusted Computing Institute at CSU. He had 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. His research interests include trusted computing, pervasive computing, mobile computing, and software engineering. He is a senior member of CCF, and a member of IEEE, ACM, and IEICE.

Jie Wu is the Chair and a Laura H. Carnell Professor in the Department of Computer and Information Sciences at Temple University, USA. Prior to joining Temple University, he was a program director at the National Science Foundation and Distinguished Professor at Florida Atlantic University. His research interests include wireless networks, mobile computing, routing protocols, fault-tolerant computing, and interconnection networks. Dr. Wu's publications include over 600 papers in scholarly journals, conference proceedings, and books. He has served on several editorial boards, including IEEE Transactions on Computers and Journal of Parallel and Distributed Computing. Dr. Wu was general co-chair for IEEE MASS-2006, IEEE IPDPS2008, and IEEE DCOSS-2009 and was the program co-chair for IEEE INFOCOMM-2011. He served as general chair for IEEE ICDCS-2013. He was an IEEE Computer Society Distinguished Visitor and the chair for the IEEE Technical Committee on Distributed Processing (TCDP). Currently, Dr. Wu is an ACM Distinguished Speaker and a Fellow of the IEEE. He is the recipient of the 2011 China Computer Federation (CCF) Overseas Outstanding Achievement Award.