Two-Dimensional Trust Rating Aggregations in Service-Oriented Applications

Yan Wang, *Senior Member*, *IEEE*, and Lei Li, *Student Member*, *IEEE*

Abstract—In most existing trust evaluation studies, a single value is computed based on the ratings given to a service provider to indicate the current trust level. This is useful but may not represent the trust features well under certain circumstances. Alternatively, a complete set of trust ratings can be transferred to a service client for local trust calculation. But this obviously creates a big overhead in communication as the data set usually has a large size covering a long service history. In this paper, we present a novel two-dimensional aggregation approach consisting of both vertical and horizontal aggregations of trust ratings. The vertical aggregation calculates the aggregated rating representing the trust level for the services delivered in a small time period. The horizontal aggregation applies our proposed optimal algorithm to determine the minimal number of time intervals, within each of which a trust vector with three values can be calculated to represent all the ratings in that time interval and retain the trust features well. Hence, a small set of trust vectors can represent a large set of trust ratings. This is significant for large-scale trust rating transmission and trust evaluation. Experiments have been conducted to illustrate the properties of our proposed approach.

Index Terms—Service trust, reputation-based trust, trust evaluation, trust rating aggregation, trust vector.

1 Introduction

In recent years, Service-Oriented Computing (SOC) has emerged to be an increasingly important research area attracting attention from both the research and industry communities [23]. In SOC applications, various services are provided to clients by different providers in a loosely coupled environment. In such context, a service can refer to a transaction, such as selling a product online (i.e., the traditional online services), or a functional component implemented by web service technologies [3]. However, when a client looks for a service out of a large pool of services provided by different service providers, in addition to functionality, the trust of service is also a key factor for service selection [26].

Conceptually, trust is the measure taken by one party on the willingness and ability of another party to act in the interest of the former party in a certain situation [17]. If the trust value is in the range of [0, 1], it can be taken as the subjective probability by which, one party expects that another party can perform a given action [13].

In SOC environments, the trust issue is very important. An effective and efficient trust management system is highly desirable and critical for service clients to identify potential risks, providing objective trust results, and preventing huge financial loss [28].

In general, in a trust management enabled system, service clients can provide feedback and trust ratings after completed transactions. A good reputation results from the high quality of delivered services in a certain time period. Based on the ratings, the trust management system can calculate the reputation-based trust value of a service provider to reflect the quality of services in a certain time period, with more weights assigned to later transactions [30], [33].

In most existing trust evaluation models [4], [16], [22], [25], [26], [27], [29], [30], [31], [32], [33], a single trust value (e.g., a value in the range of [0, 1]) is computed to reflect the global trust level of a target accumulated in a certain time period (e.g., in the latest six months). The calculation of the final trust value is based on either all the ratings given for the latest time period [16], [22], [31] or the current trust value for previous transactions and the rating for the latest transaction [27], [30].

Single trust value systems are easy to use in trust-oriented service comparison and selection. However, a single trust value computed by a service management authority cannot depict the real trust level very well under certain circumstances. For example, if there are two service providers A and B with their final trust values $T_A \approx 0.7$ and $T_B \approx 0.7$ (each of them is in the range of [0, 1]), does it mean that both A and B have the same trust level? It is not true if A's trust values are turning worse with an accumulated value of 0.7 while B's trust values are becoming better. In this case, B is better than A in terms of predicting the trust level of a forthcoming transaction. In order to observe the trend of trust changes, the complete set of trust ratings for a certain time period would be required. However, service clients are usually interested in a long service history (e.g., recent one month, three months, six months, or one year). Therefore, in such a situation, transferring a complete set of trust ratings to a client will be too costly in terms of communication overhead.

In our approach, a trust vector is computed [19],[1] which consists of three values: *final trust level (FTL)*, *service trust*

• *The authors are with the Department of Computing, Macquarie University, Sydney, NSW 2109, Australia. E-mail: {yan.wang, lei.li}@mq.edu.au.*

Manuscript received 6 May 2009; revised 14 Dec. 2009; accepted 27 May 2010; published online 25 Aug. 2010.
For information on obtaining reprints of this article, please send e-mail to: tsc@computer.org, and reference IEEECS Log Number TSCSI-2009-05-0119. Digital Object Identifier no. 10.1109/TSC.2010.39.

1. The single trust vector approach has been partially presented in our previous work published at ICWS 2008 [19].

1939-1374/11/$26.00 © 2011 IEEE Published by the IEEE Computer Society

trend (STT), and *service performance consistency level (SPCL)*. The *final trust level* is represented by a value in $[0, 1]$. The *service trust trend* is computed as a numerical value in $(-\infty, +\infty)$ representing the trend of service trust changes in a given time interval that can be interpreted as *coherent*, *upgoing*, *dropping*, or *uncertain*. The *service performance consistency level* is represented by a numerical value in [0, 1] measuring the extent to which the computed service trust trend fits the given set of trust ratings. With trust vectors, two service providers with the similar final trust values can be compared.

In the above-mentioned situations, a computed service trust vector is meaningful only if the *service performance consistency level* is high. This is because only in such a situation can the *service trust trend* represent the trend of service trust changes very well. Assuming there is a two-dimensional diagram with time t as the x-axis and rating value R as the y-axis, $R^{(t_i)}$ represents the trust rating at t_i. Given a large set of trust ratings $\{R^{(t_i)}\}$, if the trust trend changes greatly in the whole time interval $[t_{start}, t_{end}]$ ($t_i \in [t_{start}, t_{end}]$), $[t_{start}, t_{end}]$ should be divided into multiple time intervals (MTIs), each of which corresponds to a subset of ratings that can be represented by one trust vector with a high *service performance consistency level*. Thus, as all the service trust vectors cross multiple time intervals from t_{start} to t_{end}, we term the trust vector computation process and the multiple time interval analysis as the *horizontal aggregation of trust ratings*. In addition, in order to minimize the number of trust vectors, we propose an optimal multiple time interval algorithm.

Furthermore, we assume that there are a large amount of transactions for each service provider in the whole time interval $[t_{start}, t_{end}]$. Thus, if we process the ratings of all transactions occurring at different time separately, it will lead to too many trust vectors. Hence, in this paper, we take t_i as a small time period (e.g., a day) and propose a *vertical rating aggregation* approach to aggregate all the ratings $\{r_j^{(t_i)}\}$ for the services delivered during the same small time period t_i. We use $R^{(t_i)}$ to denote the aggregated rating at t_i. As time t is the horizontal axis and all ratings $\{r_j^{(t_i)}\}$ vertically distribute at the same small time period t_i, this computation process is termed as the *vertical aggregation of trust ratings*. With all the vertically aggregated ratings $\{R^{(t_i)} | R^{(t_i)}$ is the vertical aggregation of all ratings at time t_i, $t_i \in [t_{start}, t_{end}]\}$, we then apply the horizontal aggregation of trust ratings and obtain multiple trust vectors. Consequently, with both vertical and horizontal rating aggregations, a small set of trust vectors can represent a large set of trust ratings for a long service transaction history while the trust features can be highly retained.

In this paper, our contributions can be briefly summarized as follows:

1. For the vertical aggregation of trust ratings, we propose a Gaussian distribution-based analysis method and a clustering-based analysis method. The former applies to the case that all ratings conform to the Gaussian distribution, whereas the latter applies if the condition does not hold. In addition, we also propose an approach to evaluate service rating reputation (*SRR*) that can be incorporated with the above two methods to generate more objective results.
2. A trust vector with three values can depict a set of ratings better. It provides the indication of trust trend that is particularly useful in service provider comparison and selection.
3. For the horizontal aggregation of trust ratings, we propose a greedy algorithm and an optimal algorithm for generating multiple trust vectors (i.e., multiple time intervals) from a large set of trust ratings. While the greedy algorithm is usually faster, the optimal algorithm can output the minimal set of trust vectors.
4. With our proposed vertical aggregation approach and horizontal aggregation approach, a small set of values can represent a large set of trust ratings well with the trust features well retained. This is significant for large-scale trust rating transmission and trust evaluation.

This paper is organized as follows: Section 2 reviews some existing trust evaluation approaches. Section 3 presents the vertical rating aggregation approach. Section 4 introduces the service trust vector approach. In Section 5, two multiple time interval analysis algorithms are proposed. Section 6 presents our experimental studies to illustrate the properties of our proposed approach. Finally, Section 7 concludes our work.

2 Related Work

The issue of trust has been studied in some application fields.

Trust is an important issue in e-commerce (EC) environments. At eBay [1], after each transaction, a buyer can give feedback with a rating of "positive," "neutral," or "negative" to the system according to the service quality of the seller. eBay calculates the feedback score $S = P - N$, where P is the number of positive ratings left by buyers and N is the number of negative ratings. Then, the positive feedback rate $R = \frac{P}{P+N}$ (e.g., $R = 99.1\%$) is calculated and displayed on webpages. This is a simple trust management system providing valuable reputation information to buyers. In [33], Sporas system is introduced to evaluate the trust for EC applications based on the ratings of transactions in a recent time period. In this method, the ratings of later transactions are given higher weights as they are more important in trust evaluation. In [27], Wang and Lim propose an approach to evaluate the situational transaction trust, which binds the trust ratings of previous transactions with a new transaction. Since the situational trust includes service specific trust, service category trust, transaction amount category specific trust, and price trust, it can deliver more accurate trust information to buyers and prevent some typical attacks.

The trust issue has been actively studied in Peer-to-Peer (P2P) information sharing networks as a client peer needs to know prior to download actions which serving peer can provide complete files. In [4], Damiani et al. propose an approach for evaluating the reputation of peers through distributed polling algorithm and the *XRep* protocol before

initiating any download action. This approach adopts a binary rating system and it is based on the Gnutella [2] query broadcasting method. EigenTrust [16] adopts a binary rating system as well and aims to collect the local trust values of all peers to calculate the global trust value of a given peer. Some other earlier studies also adopted the binary rating system. In [31], Xiong and Liu propose a *PeerTrust* model which has two main features. First, they introduce three basic trust parameters (i.e., the feedback that a peer receives from other peers, the total number of transactions that a peer performs, and the credibility of the feedback sources) and two adaptive factors in computing trustworthiness of peers (i.e., transaction context factor and the community context factor). Second, they define some general trust metrics and formulas to aggregate these parameters into a final trust value.

In the literature, the issue of trust has also received much attention in the field of service-oriented computing. In [26], Vu et al. present a model to evaluate the service trust by comparing the advertised service quality and the delivered service quality. If the advertised service quality is as good as the delivered service quality, the service is reputable. In [22], Lin et al. propose a distributed trust evaluation framework that consists of trust brokers and reputation authorities, and consider the aggregation of trust values from different brokers and authorities. In [29], Wang et al. propose some trust evaluation metrics and a formula for trust computation, with which a final trust value is computed. In addition, they propose a fuzzy logic-based approach for determining reputation ranks that particularly differentiate the service periods of new service providers and old (long-existing) ones. The aim is to provide incentives to new service providers and penalize those old service providers with poor service quality.

Trust also has drawn much attention in the field of multiagent systems. In [12], Jøsang describes a framework for combining and assessing subjective ratings from different sources based on Dempster-Shafer belief theory. In [25], Teacy et al. introduce the Trust and Reputation model for Agent-based Virtual OrganisationS (TRAVOS) system which calculates an agent's trust on an interaction partner using probability theory, taking into account the past interactions between agents. In [7], Griffiths proposes a multidimensional trust model which allows agents to model the trust value of others according to various criteria. In [11], the proposed model addresses the trust issue from the perspective of multiagent systems where in addition to trust evaluation, motivations of agents and the dependency relationships among them are also taken into account. In [5], a community-wide trust evaluation method is proposed where the final trust value is computed by aggregating the ratings (termed as votes in [5]) and other aspects (e.g., the rater's location and connection medium). In addition, this approach computes the trust level of an assertion (e.g., trustworthy, untrustworthy) as the aggregation of multiple fuzzy values representing the trust resulting from human interactions.

Similar to the taxonomy in [5], we categorize the above trust aggregation approaches as follows according to their computation techniques. Some models may correlate to more than one category. *Category 1* adopts the approach to calculating the summation or weighted average of ratings, like the models in [5], [7], [27], [29], [31], and [33]. A few studies address how to compute the final trust value by considering appropriate metrics. For example, later transactions are more important [33]; the evaluation approach should provide incentive to consistently good quality services and punish malicious service providers [29], [31]. Some other studies also consider context factors, e.g., the new transaction amount and service category [27], the rater's profile and location [5], or the relationship between the rater's group and the ratee [7]. *Category 2* addresses the subjective property of trust for trust rating aggregation, e.g., the work in [12] and [20], where the subjective probability theory is adopted in trust evaluation. The approaches in *Category 3* (e.g., [25]) adopt Bayesian systems, which take binary ratings as input and compute reputation scores by statistically updating beta probability density functions (PDF). *Category 4* uses flow models, e.g., in [33], which compute the trust of a target through some intermediate participants and the trust dependency between them. While each of the above categories calculates a crisp value, *the last category* adopts fuzzy models, e.g., in [5] and [29], where membership functions are used to determine the trustworthiness of targets.

The above-mentioned approaches contribute to trust evaluation by incorporating various aspects and factors. An aggregated single trust value is computed to reflect the global trust level and is easy to use in service provider comparison and selection. Nevertheless, as we have mentioned in Section 1, the single trust value may not depict the trust level well under some circumstances. Different from existing single value-based trust evaluation models, we propose a novel two-dimensional approach consisting of both vertical and horizontal aggregations of trust ratings. It is based on our previous work on single trust vector evaluation [19]. In [19], a trust vector consists of three values: Final Trust Value (*FTL*), Service Trust Trend (*STT*), and Service Performance Consistency Level (*SPCL*). While the *FTL* value is similar to the result of some other single trust value approaches, together with *STT* and *SPCL*, a trust vector can better depict the trust features of a large set of trust ratings and is useful for trustworthy service provider selection. In Section 4, we briefly introduce the single trust vector approach. In this paper, we extend our previous work in two major directions. On one hand, a set of ratings in a small time period (e.g., a day) are aggregated yielding a single rating representing the trust level for that time period. On the other hand, multiple trust vectors may have to be generated for the whole time interval of service history so that each trust vector can represent the trust ratings in its time interval precisely. The ultimate purpose of our work is to represent the large volume of trust ratings with a small data set while the trust features are well depicted and retained.

In the literature, there exist some other approaches using trust vectors, but with different focuses. In [24], Ray and Chakraborty propose a trust vector that consists of experience, knowledge, and recommendation. The focus is how to address these three independent aspects of trust, as listed in a trust vector. Therefore, its goal is different from ours. In [34], Zhao and Li propose a method using a trust vector to represent the directed link with a trust value between two

peers. A trust vector includes a truster, a trustee, and the trust value between them. Thus, their trust vector is totally different from our approach.

3 Vertical Aggregation of Trust Ratings

In this section, we introduce our proposed vertical rating aggregation approach, which aggregates the ratings $\{r_j^{(t_i)}|j = 1,\ldots,m\}$ for the services delivered at a small time period t_i (e.g., a day) to a single trust value $R^{(t_i)}$.

3.1 Vertical Aggregation without Service Rating Reputation

In order to aggregate the ratings $\{r_j^{(t_i)}\}$ vertically, we first consider the distribution of these ratings given by different clients. Let $\bar{\mathbf{r}}^{(t_i)}$ denote the rating that would ideally represent the trust level of the service delivered at the time period t_i. If most clients are honest, then their ratings are close to $\bar{\mathbf{r}}^{(t_i)}$. Raters of these ratings are taken as the main stream. Thus, each rating with a clear distance to $\bar{\mathbf{r}}^{(t_i)}$ is taken as marginal. As pointed out in cognitive science, in the process of decision making, the cognitive and personal preference is usually needed to be taken into account [18]. One of the most commonly existing cognitive preferences is a willingness to believe what we have been told most often and by the greatest number of different sources [18]. Following this principle of cognitive science, marginal ratings can be identified and discarded. If we can determine the upper control limit $R_u^{(t_i)}$ and the lower control limit $R_l^{(t_i)}$ properly, marginal ratings out of the range $[R_l^{(t_i)}, R_u^{(t_i)}]$ can be filtered out. Then, an aggregated rating $R^{(t_i)}$ can be derived from the rest ratings in $[R_l^{(t_i)}, R_u^{(t_i)}]$ and taken as the *estimation* of $\bar{\mathbf{r}}^{(t_i)}$.

In this section, we propose a *Gaussian distribution-based analysis method* and a *Clustering-based analysis method* to compute the aggregated rating $R^{(t_i)}$ in different situations.

3.1.1 Gaussian Distribution-Based Analysis Method

As illustrated in [10], if all service clients give feedback after transactions, the provided ratings conform to the Gaussian distribution. A complete set of honest ratings can be collected based on honest-feedback-incentive mechanisms [14], [15]. Therefore, the computation method introduced in this section applies to the case that the ratings given for the services delivered at t_i can approximately conform to the Gaussian distribution. In order to determine if ratings conform to the Gaussian distribution, we adopt the formal goodness-of-fit testing procedure based on the chi-square distribution [9]. If ratings do not conform to the Gaussian distribution, we will adopt our clustering-based analysis method for vertical aggregation.

If ratings conform to the Gaussian distribution, according to the empirical rule in statistics [9], about 95.45 percent values in the Gaussian distribution are within $[\mu - 2\sigma, \mu + 2\sigma]$ where σ is the standard deviation and μ is the mean. Hence, based on the control chart in the statistical quality control [9], we can adopt $\mu + 2\sigma$ as the upper control limit $R_u^{(t_i)}$ and adopt $\mu - 2\sigma$ as the lower control limit $R_l^{(t_i)}$.

Definition 1. *Based on the unbiased estimation [9], the centerline $R_c^{(t_i)}$, the upper control limit $R_u^{(t_i)}$, and the lower control limit $R_l^{(t_i)}$ of trust ratings delivered at the small time period t_i are defined in sequence as follows:*

$$R_c^{(t_i)} = \frac{1}{m}\sum_{j=1}^{m} r_j^{(t_i)}, \tag{1}$$

$$R_u^{(t_i)} = R_c^{(t_i)} + 2\sqrt{\frac{\sum_{j=1}^{m}(r_j^{(t_i)} - R_c^{(t_i)})^2}{m-1}}, \tag{2}$$

$$R_l^{(t_i)} = R_c^{(t_i)} - 2\sqrt{\frac{\sum_{j=1}^{m}(r_j^{(t_i)} - R_c^{(t_i)})^2}{m-1}}, \tag{3}$$

where $r_j^{(t_i)} \in [0,1]$ is the trust rating from client j ($j = 1,\ldots,m$) for a service delivered at time t_i ($i = 1,\ldots,n$).

So, the trust ratings out of the range $[R_l^{(t_i)}, R_u^{(t_i)}]$ are taken as marginal ratings.

Definition 2. *If there are m' trust ratings $\{r_k^{(t_i)'}\}$ in the range of $[R_l^{(t_i)}, R_u^{(t_i)}]$, the vertically aggregated rating $R^{(t_i)}$ can be calculated by*

$$R^{(t_i)} = \frac{1}{m'}\sum_{k=1}^{m'} r_k'^{(t_i)}. \tag{4}$$

3.1.2 Clustering-Based Analysis Method

If ratings do not conform to the Gaussian distribution, the clustering-based analysis method will be applied.

In this method, we adopt the hierarchical clustering method [8], which creates a hierarchical cluster of the given data set by either clustering from one cluster or n_r clusters (n_r is the size of the data set) until all clusters become stable. In order to determine marginal rating clusters, the *divisive hierarchical clustering approach* [8] is selected, which starts the decomposition from one cluster.

Before presenting the divisive hierarchical clustering approach, we first introduce some definitions.

The *relative rating density* from $r_h^{(t_i)}$ to $r_j^{(t_i)}$ at the small time period t_i is

$$D_r^{(t_i)}\left(r_h^{(t_i)}, r_j^{(t_i)}\right) = \frac{\sum_{r_h^{(t_i)} < r_k^{(t_i)} < r_j^{(t_i)}} fre(r_k^{(t_i)})}{r_j^{(t_i)} - r_h^{(t_i)}}, \tag{5}$$

where $r_h^{(t_i)}$ and $r_j^{(t_i)}$ ($r_h^{(t_i)} < r_j^{(t_i)}$) are ratings, and $fre(r_k^{(t_i)})$ is the frequency of $r_k^{(t_i)}$.

The *marginal rating percentage* at the small time period t_i is

$$P_{marginal}^{(t_i)} = \frac{n_{marginal}^{(t_i)}}{n_{total}^{(t_i)}}, \tag{6}$$

where $n_{marginal}^{(t_i)}$ is the number of marginal ratings delivered at the small time period t_i and $n_{total}^{(t_i)}$ is the total number of ratings for the services delivered at the small time period t_i.

The divisive hierarchical clustering approach works as follows: initially, all the ratings are placed in one cluster, and the centerline $R_c^{(t_i)}$ is calculated with all ratings by (1). Then, the cluster is split according to relative rating density in the cluster. This process repeats until $D_r^{(t_i)}$ or $P_{marginal}^{(t_i)}$ reaches the corresponding threshold (such as $D_r^{(t_i)} = 0.10$ and $P_{marginal}^{(t_i)} = 0.10$). All ratings that are not in the centered

cluster are taken as marginal ratings. In the centered cluster, the two boundary ratings are used as $R_l^{(t_i)}$ and $R_u^{(t_i)}$. With all ratings in $[R_l^{(t_i)}, R_u^{(t_i)}]$, the vertically aggregated rating $R^{(t_i)}$ can be computed according to (4).

3.2 Vertical Aggregation with Service Rating Reputation

The client's rating reputation is important for estimating the ideal rating $\bar{\mathbf{r}}^{(t_i)}$. This reputation can be evaluated with the distance from the client's rating to $R^{(t_i)}$. Obviously, the smaller the distance, the better the service rating reputation (*SRR*). With the *SRR* of all ratings, $R^{(t_i)}$ can be recalculated. This leads to an iterative process until all computed values become stable.

3.2.1 Service Rating Reputation Evaluation

The calculation of *SRR* follows a common principle as follows, which appears in a number of studies [19], [21], [30], [33].

Principle 1. *The reputation value is computed by taking the service trust value in a recent time period into account with higher weights given to the ratings of later services.*

Definition 3. *$V_{SRR_j}^{(t_i)}$, the SRR value for client j from t_1 to t_i, can be calculated as follows:*

$$V_{SRR_j}^{(t_i)} = \frac{\sum_{k=1}^{i} w_{t_k} \cdot R_{SRR_j}^{(t_k)}}{\sum_{k=1}^{i} w_{t_k}}, \tag{7}$$

where $R_{SRR_j}^{(t_k)}$ is the SRR value for client j at the small time period t_k ($k = 1, \ldots, i$), and w_{t_k} is the weight for $R_{SRR_j}^{(t_k)}$, which can be calculated as the exponential moving average [9]:

$$w_{t_k} = \alpha^{n-k}, \quad 0 < \alpha \leq 1. \tag{8}$$

In addition, because the smaller the distance between a rating for a client and the estimation of $\bar{\mathbf{r}}^{(t_i)}$, the bigger and better the *SRR*, and vice versa, a principle about the $R_{SRR_j}^{(t_i)}$ evaluation is introduced as follows:

Principle 2. *$R_{SRR_j}^{(t_i)}$, the SRR value for client j at the small time period t_i ($i = 1, \ldots, n$), is a monotonically decreasing function of the distance*

$$r_{j_{dis}}^{(t_i)} = \left| r_j^{(t_i)} - R_c^{(t_i)'} \right|, \tag{9}$$

where $r_j^{(t_i)}$ is the trust rating from client j for the service delivered at the small time period t_i, $R_c^{(t_i)'}$ is the weighted average of $r_j^{(t_i)}$, and $V_{SRR_j}^{(t_i)}$ is the weight defined in Definition 3.

According to Principle 2, $R_{SRR_j}^{(t_i)}$ can be calculated by the following formula.

Definition 4. *$R_{SRR_j}^{(t_i)}$, the SRR value for client j at the small time period t_i ($i = 1, \ldots, n$), can be evaluated as*

$$R_{SRR_j}^{(t_i)} = \begin{cases} 1 - 2^{2m_0-1}\left(\frac{r_{j_{dis}}^{(t_i)}}{\gamma}\right)^{2m_0}, & \text{if } 0 \leq r_{j_{dis}}^{(t_i)} \leq \frac{\gamma}{2}, \\ 2^{2m_0-1}\left(\frac{r_{j_{dis}}^{(t_i)}}{\gamma} - 1\right)^{2m_0}, & \text{if } \frac{\gamma}{2} < r_{j_{dis}}^{(t_i)} \leq \gamma, \end{cases} \tag{10}$$

where $\gamma = \max r_{j_{dis}}^{(t_i)}$, and m_0 is the argument to control the function curve.

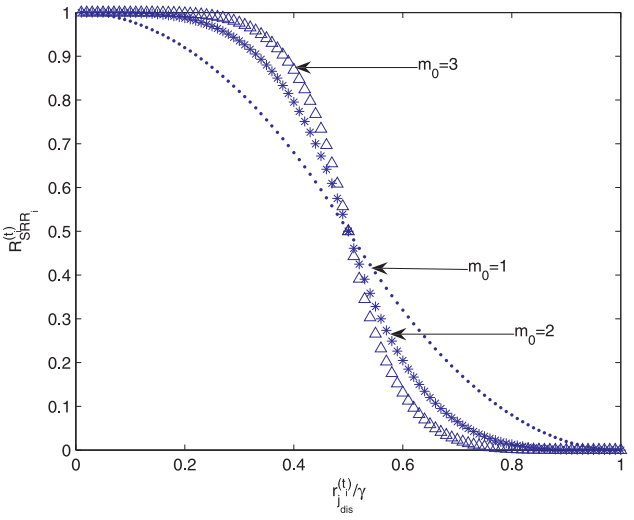

Fig. 1. $R_{SRR_j}^{(t_i)}$ function.

When setting $m_0 = 1, 2$, or 3, the changes of the function curve in (10) are depicted in Fig. 1. It is easy to see that in all cases, $R_{SRR_j}^{(t_i)}$ is the monotonically decreasing function of $r_{j_{dis}}^{(t_i)}$, following Principle 2.

3.2.2 Vertical Aggregation of Trust Ratings

With *SRR* taken into account, we should refine the Gaussian distribution-based analysis method and the clustering-based analysis method.

Definition 5. *Based on the weighted unbiased estimation [9], the centerline $R_c^{(t_i)'}$, the upper control limit $R_u^{(t_i)'}$, and the lower control limit $R_l^{(t_i)'}$ can be calculated in sequence as follows:*

$$R_c^{(t_i)'} = \frac{\sum_{j=1}^{n} V_{SRR_j}^{(t_i)} \cdot r_j^{(t_i)}}{\sum_{j=1}^{n} V_{SRR_j}^{(t_i)}}, \tag{11}$$

$$R_u^{(t_i)'} = R_c^{(t_i)'} + 2\sqrt{\frac{\sum_{j=1}^{n} V_{SRR_j}^{(t_i)} \cdot (r_j^{(t_i)} - R_c^{(t_i)'})^2}{\sum_{j=1}^{n} V_{SRR_j}^{(t_i)} - 1}}, \tag{12}$$

$$R_l^{(t_i)'} = R_c^{(t_i)'} - 2\sqrt{\frac{\sum_{j=1}^{n} V_{SRR_j}^{(t_i)} \cdot (r_j^{(t_i)} - R_c^{(t_i)'})^2}{\sum_{j=1}^{n} V_{SRR_j}^{(t_i)} - 1}}, \tag{13}$$

where $r_j^{(t_i)}$ is the trust rating from client j for the service delivered at the small time period t_i ($i = 1, \ldots, n$), and $V_{SRR_j}^{(t_i)}$ is the service rating reputation value for client j defined in (7).

Definition 6. *If there are m'' trust ratings $\{r_k^{(t_i)''}\}$ in the range of $[R_l^{(t_i)'}, R_u^{(t_i)'}]$ and $R_{SRR_k}^{(t_i)} > \epsilon_3$, the vertically aggregated rating $R^{(t_i)'}$ can be defined as*

$$R^{(t_i)'} = \frac{\sum_{k=1}^{m''} V_{SRR_k}^{(t_i)} \cdot r_k^{(t_i)''}}{\sum_{k=1}^{m''} V_{SRR_k}^{(t_i)}}, \tag{14}$$

where $R_u^{(t_i)'}$ is the upper control limit defined in (12), $R_l^{(t_i)'}$ is the lower control limit defined in (13), and ϵ_3 ($\epsilon_3 \in [0, 1]$) is a threshold.

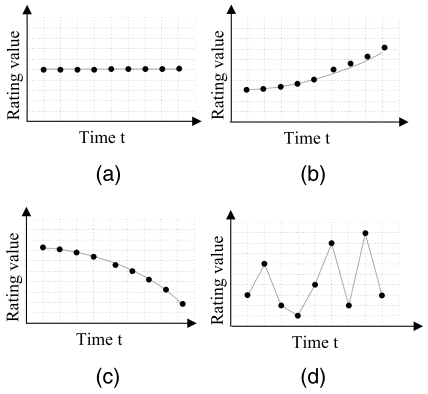

Fig. 2. Several *STT* cases. (a) Coherent. (b) Upgoing. (c) Dropping. (d) Uncertain.

As for the clustering-based analysis method, all the processes are the same as the method introduced in Section 3.1.2 except that *SRR* is added in computation.

In order to evaluate the *SRR*, the centerline $R_c^{(t_i)'}$ should be known. However, from (11), $V_{SRR_j}^{(t_i)}$ is necessary for determining $R_c^{(t_i)'}$. Therefore, it is an iterative process to compute $R_c^{(t_i)'}$ and $V_{SRR_j}^{(t_i)}$.

Here, we take the Gaussian distribution-based analysis method as an example to illustrate the iterative vertical rating aggregation process in Algorithm 1. Obviously, the iterative process is similar in the vertical rating aggregation on top of the clustering-based analysis method.

Algorithm 1. Vertical rating aggregation algorithm
Input: trust ratings $r_j^{(t_i)}$,
an arbitrary small positive number ϵ (such as 0.0001).
Output: $V_{SRR_j}^{(t_i)}$, $R^{(t_i)'}$.

1: Initialize $R_{SRR_j}^{(t_i)}$ by (10)
2: $r_{i_{a_1}} \Leftarrow 0$
3: $r_{i_{a_2}} \Leftarrow 0$
4: **while** $\max|r_{i_{a_1}} - r_{i_{a_2}}| > \epsilon$ **do**
5: $\quad r_{i_{a_2}} \Leftarrow r_{i_{a_1}}$
6: $\quad$ compute $V_{SRR_j}^{(t_i)}$ with $R_{SRR_j}^{(t_i)}$ by (7)
7: $\quad$ compute $r_{i_{a_1}}$ by (11)
8: $\quad$ compute $R_{SRR_j}^{(t_i)}$ by (10)
9: **end while**
10: $R_c^{(t_i)'} \Leftarrow r_{i_{a_1}}$
11: compute $R_u^{(t_i)'}$ by (12)
12: compute $R_l^{(t_i)'}$ by (13)
13: compute $R^{(t_i)'}$ by (14)

In summary, the main difference between the vertically aggregated rating evaluation introduced in Section 3.1 and the one in this section is that the latter approach takes *SRR* into account, which leads to more objective results.

4 Service Trust Vector

In this section, the trust vector approach is presented. A trust vector is used to depict the trust level with three values including *Final Trust Level*, *Service Trust Trend*, and *Service Performance Consistency Level*. The trust vector approach can be applied to the set of ratings $\{R^{(t_i)}\}$ obtained from the vertical aggregation introduced in Section 3.

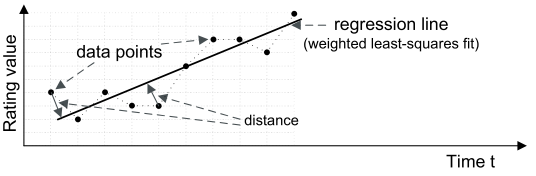

Fig. 3. Weighted least-squares linear regression.

4.1 Final Trust Level Evaluation

Based on Principle 1, the Final Trust Level (*FTL*) value for the time interval $[t_{start}, t_{end}]$ can be calculated as

$$V_{FTL}^{[t_{start},t_{end}]} = \frac{\sum_{i=1}^{n} w_{t_i} \cdot R^{(t_i)}}{\sum_{i=1}^{n} w_{t_i}}, \tag{15}$$

where $t_i \in [t_{start}, t_{end}]$ and w_{t_i} is the weight defined in (8).

Actually, any existing single trust value method can be adopted to compute the *FTL* value if it is based on all nonbinary ratings $\{R^{(t_i)}\}$, e.g., the methods proposed in [29] and [33].

4.2 Service Trust Trend Evaluation

Service Trust Trend (*STT*) aims to illustrate the trend of service trust value changes in a given time interval. Some typical cases of *STT* are depicted in Fig. 2, which are "*coherent*," "*upgoing*," "*dropping*," and "*uncertain*" in sequence.

In order to evaluate the *STT* of a set of ratings $\{R^{(t_i)}|t_i \in [t_{start}, t_{end}]\}$ for the time interval $[t_{start}, t_{end}]$, following Principle 1, we design a *weighted least-squares linear regression* method [19], as illustrated in Fig. 3. This method is used to obtain the best-fit straight line from a set of data points. It is characterized by the sum of weighted squared residuals with its least value, where a residual is the distance from a data point to the regression line (see Fig. 3). Once the regression line is obtained, its slope is the *STT* value.

Let $(t_1, R^{(t_1)}), (t_2, R^{(t_2)}), \ldots, (t_n, R^{(t_n)})$ be the given data points within time interval $[t_{start}, t_{end}]$, where $R^{(t_i)} \in [0, 1]$ is the trust rating delivered at the small time period t_i ($t_i < t_{i+1}$, $t_1 = t_{start}$, and $t_n = t_{end}$). In general, $R^{(t_i)}$ can be the rating at t_i aggregated by the vertical aggregation approach introduced in Sections 3.1 or 3.2.

The *sum of squares of the distance* from point $(t_i, R^{(t_i)})$ to the regression line $R = a_0 + a_1 t$ can be calculated as follows:

$$S = \sum_{i=1}^{n} {w_{t_i}}^2 d_i^2 = \sum_{i=1}^{n} \frac{{w_{t_i}}^2 (R^{(t_i)} - a_0 - a_1 t_i)^2}{1 + a_1^2}. \tag{16}$$

Now, the task is to minimize the *sum of squares of the distance* S with respect to the parameters a_0 and a_1. Hence, we differentiate S with respect to a_0 and a_1, and set the results to zero, yielding

$$a_1^2 + \frac{S_{wr2} S_w - S_{wr}^2 + S_{wt}^2 - S_{wt2} S_w}{S_{wr} S_{wt} - \sum_{i=1}^{n} w_{t_i}^2 t_i R^{(t_i)} S_w} a_1 - 1 = 0, \tag{17}$$

where $S_w = \sum_{i=1}^{n} w_{t_i}^2$, $S_{wt} = \sum_{i=1}^{n} w_{t_i}^2 t_i$, $S_{wr} = \sum_{i=1}^{n} w_{t_i}^2 R^{(t_i)}$, $S_{wt2} = \sum_{i=1}^{n} w_{t_i}^2 t_i^2$, and $S_{wr2} = \sum_{i=1}^{n} w_{t_i}^2 {R^{(t_i)}}^2$. Obviously, it is easy to obtain the very small real solution of a_1 in (17).

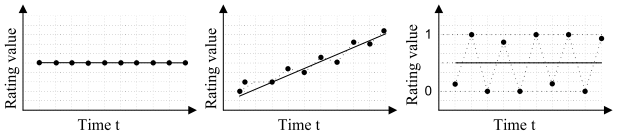

Fig. 4. Several *SPCL* cases. (a) Absolutely consistent. (b) Relatively consistent. (c) Inconsistent.

Hence, based on the weighted least-squares linear regression method, we can obtain the *STT* value $V_{STT} = a_1$ from (17). However, in order to determine the four cases of *STT*, another factor *SPCL* should be taken into account.

4.3 Service Performance Consistency Level Evaluation

The Service Performance Consistency Level (*SPCL*) value indicates the consistency level of the service trust values in a certain time interval. Some typical *SPCL* cases are depicted in Fig. 4. In sequence, they are "absolutely consistent," "relatively consistent," and "inconsistent."

Prior to presenting the detailed *SPCL* evaluation method, we first introduce Definition 7.

Definition 7. *Following Principle 1, the* weighted average distance *for the time interval* $[t_{start}, t_{end}]$ *is*

$$v_{t_{dis}}^{[t_{start},t_{end}]} = \frac{\sum_{i=1}^{n} w_{t_i} |R^{(t_i)} - (a_0 + a_1 t_i)|}{\sqrt{1 + a_1^2} \sum_{i=1}^{n} w_{t_i}} \tag{18}$$

for n *trust ratings* $\{R^{(t_i)}\}$ *delivered in the time interval* $[t_{start}, t_{end}]$.

$V_{SPCL}^{[t_{start},t_{end}]}$, the *SPCL* value for the time interval $[t_{start}, t_{end}]$, is a monotonically decreasing function of the weighted mean distance $v_{t_{dis}}^{[t_{start},t_{end}]}$.

Definition 8. *The SPCL value for the time interval* $[t_{start}, t_{end}]$ *is*

$$\begin{aligned} V_{SPCL}^{[t_{start},t_{end}]} &= 1 - 2v_{t_{dis}}^{[t_{start},t_{end}]} \\ &= 1 - 2\frac{\sum_{i=1}^{n} w_{t_i} |R^{(t_i)} - (a_0 + a_1 t_i)|}{\sqrt{1 + a_1^2} \sum_{i=1}^{n} w_{t_i}}. \end{aligned} \tag{19}$$

With V_{SPCL} and V_{STT}, the four cases of *STT* can be determined.

1. If $V_{SPCL} > \epsilon_4$ and $|V_{STT}| < \epsilon_5$ ($0 < \epsilon_5 \ll 1$ is a threshold), *STT* is *coherent* (see Fig. 2a), i.e., the service trust value remains at the same level.
2. If $V_{SPCL} > \epsilon_4$ and $V_{STT} > \epsilon_5$, *STT* is *upgoing* (see Fig. 2b), i.e., the service trust value is becoming better.
3. If $V_{SPCL} > \epsilon_4$ and $V_{STT} < -\epsilon_5$, *STT* is *dropping* (see Fig. 2c), i.e., the service trust value is turning worse.
4. If $V_{SPCL} < \epsilon_4$, *STT* is *uncertain* (see Fig. 2d), i.e., the service trust value is not reliable.

4.4 Service Trust Vector

Based on the above discussions, we can define the service trust vector as follows:

Definition 9. *The* service trust vector $\overline{T}^{[t_{start},t_{end}]}$ *is*

$$\overline{T}^{[t_{start},t_{end}]} = <V_{FTL}^{[t_{start},t_{end}]}, V_{STT}^{[t_{start},t_{end}]}, V_{SPCL}^{[t_{start},t_{end}]}>, \tag{20}$$

where $V_{FTL}^{[t_{start},t_{end}]}$ *is defined in (15),* $V_{STT}^{[t_{start},t_{end}]}$ *is decided by (17), and* $V_{SPCL}^{[t_{start},t_{end}]}$ *is defined in (19).*

With trust vectors, all service providers form a partial order set. Given two service providers P_j and P_k with their service trust vectors

$$\overline{T}_j^{[t_{start},t_{end}]} = <V_{FTL_j}^{[t_{start},t_{end}]}, V_{STT_j}^{[t_{start},t_{end}]}, V_{SPCL_j}^{[t_{start},t_{end}]}>$$

and

$$\overline{T}_k^{[t_{start},t_{end}]} = <V_{FTL_k}^{[t_{start},t_{end}]}, V_{STT_j}^{[t_{start},t_{end}]}, V_{SPCL_k}^{[t_{start},t_{end}]}>,$$

they are comparable in the following cases:

Property 1. *If* $V_{STT_j}^{[t_{start},t_{end}]} = V_{STT_k}^{[t_{start},t_{end}]}$, $V_{SPCL_j}^{[t_{start},t_{end}]} = V_{SPCL_k}^{[t_{start},t_{end}]}$, *and* $V_{FTL_j}^{[t_{start},t_{end}]} < V_{FTL_k}^{[t_{start},t_{end}]}$, P_k *is more preferable, which is denoted as* $P_k > P_j$ *or* $P_j < P_k$.

Property 2. *If* $V_{FTL_j}^{[t_{start},t_{end}]} = V_{FTL_k}^{[t_{start},t_{end}]}$, $V_{SPCL_j}^{[t_{start},t_{end}]} = V_{SPCL_k}^{[t_{start},t_{end}]}$, *and* $V_{STT_j}^{[t_{start},t_{end}]} < V_{STT_k}^{[t_{start},t_{end}]}$, *then* $P_j < P_k$.

Property 3. *If* $V_{FTL_j}^{[t_{start},t_{end}]} = V_{FTL_k}^{[t_{start},t_{end}]}$, $V_{STT_j}^{[t_{start},t_{end}]} = V_{STT_k}^{[t_{start},t_{end}]}$, *and* $V_{SPCL_j}^{[t_{start},t_{end}]} < V_{SPCL_k}^{[t_{start},t_{end}]}$, *then* $P_j < P_k$.

In addition, when the two values in a service vector element are approximately equal, we can compare the two trust vectors in the following cases:

Property 4. *If*

$$|V_{STT_j}^{[t_{start},t_{end}]} - V_{STT_k}^{[t_{start},t_{end}]}| < \epsilon_6, \tag{21}$$

$$|V_{SPCL_j}^{[t_{start},t_{end}]} - V_{SPCL_k}^{[t_{start},t_{end}]}| < \epsilon_7, \tag{22}$$

and

$$|V_{FTL_j}^{[t_{start},t_{end}]} - V_{FTL_k}^{[t_{start},t_{end}]}| < \epsilon_8, \tag{23}$$

P_j *and* P_k *are both preferable, which is denoted as* $P_k = P_j$, *where* $0 < \epsilon_6, \epsilon_7, \epsilon_8 \ll 1$ *are thresholds that can be specified by service clients or trust management authorities.*

Property 5. *If*

$$|V_{STT_j}^{[t_{start},t_{end}]} - V_{STT_k}^{[t_{start},t_{end}]}| < \epsilon_6, \tag{24}$$

$$|V_{SPCL_j}^{[t_{start},t_{end}]} - V_{SPCL_k}^{[t_{start},t_{end}]}| < \epsilon_7, \tag{25}$$

and

$$V_{FTL_j}^{[t_{start},t_{end}]} + \epsilon_8 < V_{FTL_k}^{[t_{start},t_{end}]}, \tag{26}$$

P_k *is more preferable, which is denoted as* $P_k > P_j$ *or* $P_j < P_k$.

Property 6. *If*

$$|V_{FTL_j}^{[t_{start},t_{end}]} - V_{FTL_k}^{[t_{start},t_{end}]}| < \epsilon_8, \tag{27}$$

$$|V_{SPCL_j}^{[t_{start},t_{end}]} - V_{SPCL_k}^{[t_{start},t_{end}]}| < \epsilon_7, \tag{28}$$

and

$$V_{STT_j}^{[t_{start},t_{end}]} + \epsilon_6 < V_{STT_k}^{[t_{start},t_{end}]}, \tag{29}$$

then $P_j < P_k$.

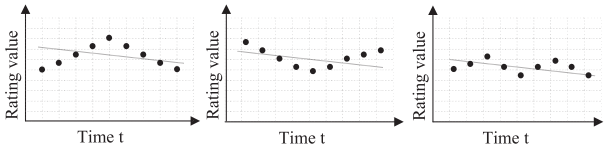

Fig. 5. Several MTI examples.

Property 7. *If*

$$|V_{FTL_j}^{[t_{start},t_{end}]} - V_{FTL_k}^{[t_{start},t_{end}]}| < \epsilon_8, \tag{30}$$

$$|V_{STT_j}^{[t_{start},t_{end}]} - V_{STT_k}^{[t_{start},t_{end}]}| < \epsilon_6, \tag{31}$$

and

$$V_{SPCL_j}^{[t_{start},t_{end}]} + \epsilon_7 < V_{SPCL_k}^{[t_{start},t_{end}]}, \tag{32}$$

then $P_j < P_k$.

5 Multiple Time Interval Analysis

A single trust vector with three values can represent the ratings in the given interval $[t_1, t_n]$ well if its *SPCL* value is high (i.e., 0.9 or more). A high *SPCL* value indicates that all rating points are very close the obtained regression line. However, if the trust trend significantly changes in $[t_1, t_n]$, though a single trust vector can be computed, the *SPCL* value will be low indicating that the vector cannot depict all the ratings precisely. In this case, in order to represent all trust ratings better, multiple intervals in $[t_1, t_n]$ should be determined, within each of which one trust vector with a high *SPCL* value can be obtained to represent all corresponding ratings with a consistent trust trend.

Fig. 5 illustrates three cases. We can notice that these cases are quite different from each other in terms of trust trend changes. If only one trust vector is computed in each case, all three cases will have approximately the same V_{FTL}, V_{STT}, and V_{SPCL}. Meanwhile, in each case, most points have clear distances to the obtained regression line. This leads to a low *SPCL* value indicating the obtained single trust vector cannot represent all the trust ratings well.

In this section, we develop the Multiple Time Interval (MTI) algorithms to find the set of regression lines from the staring point $(t_1, R^{(t_1)})$ to the ending point $(t_n, R^{(t_n)})$. A regression line may exist from $(t_i, R^{(t_i)})$ to $(t_j, R^{(t_j)})(i < j)$ only if the corresponding $V_{SPCL}^{[t_i,t_j]} \geq \epsilon_9 (0 \ll \epsilon_9 < 1$ is a threshold, e.g., $\epsilon_9 = 0.9)$. For this purpose, we propose a greedy algorithm and an optimal algorithm for the MTI analysis. The latter algorithm can return a set of MTI with the minimal number of intervals.

5.1 Greedy MTI Algorithm

The greedy algorithm (Algorithm 2) works as follows:

- Step 1. Take $(t_1, R^{(t_1)})$ as the staring point and $(t_n, R^{(t_n)})$ as the first ending point. If $V_{SPCL}^{[t_1,t_n]} \geq \epsilon_9$, the regression line starts from $(t_1, R^{(t_1)})$ and ends at $(t_n, R^{(t_n)})$; otherwise, $(t_{n-1}, R^{(t_{n-1})})$ is taken as the ending point for testing if $V_{SPCL}^{[t_1,t_{n-1}]} \geq \epsilon_9$ and so forth. Thus, the first ending point $(t_i, R^{(t_i)})(i = 2, \ldots, n)$ can be determined so that $V_{SPCL}^{[t_1,t_i]} \geq \epsilon_9$. By doing so, the obtained regression line is the longest one staring from $(t_1, R^{(t_1)})$ (lines 5-7 in Algorithm 2).
- Step 2. Taking $(t_i, R^{(t_i)})$ as the new staring point, the same process can repeat so that a regression line can be drawn from $(t_i, R^{(t_i)})$ to $(t_j, R^{(t_j)})(j = i+1, \ldots, n)$ with $V_{SPCL}^{[t_i,t_j]} \geq \epsilon_9$ (lines 5-12).
- Step 3. The above process repeats until the last regression line reaches point $(t_n, R^{(t_n)})$.

Algorithm 2. Greedy MTI algorithm
Input: trust ratings $R^{(t_i)}$,
the given interval $[t_1, t_n]$,
the threshold ϵ_9 of V_{SPCL} (such as 0.9).
Output: MTI boundary $t_{lb_j} t_{rb_j}$.

```
1: j ⇐ 1
2: left time boundary t_lbj ⇐ t_1
3: right time boundary t_rbj ⇐ t_n
4: while V_SPCL^[t_lbj, t_n] < ε9 do
5:     while V_SPCL^[t_lbj, t_rbj] < ε9 do
6:         t_rbj ⇐ t_rbj − small time period unit
7:     end while
8:     if t_rbj ≠ t_n then
9:         j ⇐ j + 1
10:        t_lbj ⇐ t_rbj−1
11:        t_rbj ⇐ t_n
12:    end if
13: end while
```

As the computation of each trust vector element has a complexity of $O(n)$, the greedy MTI algorithm incurs a complexity of $O(n^3)$ and may not find a set of MTI with the minimal number of intervals.

5.2 Optimal MTI Algorithm

Now, we introduce an optimal MTI algorithm which can deliver the minimal number of regression lines. In this algorithm, each point $(t_i, R^{(t_i)})$ is taken as a vertex v_i in a graph. There is a directed edge between v_i and $v_j (i < j)$ of weight 1 if $V_{SPCL}^{[t_i,t_j]} \geq \epsilon_9$. Thus, the task to obtain a set of MTI with the minimal number of intervals is converted to the one to find the shortest path from v_1 (i.e., point $(t_1, R^{(t_1)})$) to v_n (i.e., point $(t_n, R^{(t_n)})$). For this task, we extend Dijkstra's shortest path algorithm [6]. The shortest path from v_1 to v_n corresponds to the set of regression lines from $(t_1, R^{(t_1)})$ to $(t_n, R^{(t_n)})$ with the minimum number of lines.

The optimal algorithm (Algorithm 3) works as follows:

- Step 1. Initialize the adjacent matrix with n vertices where the weight of the edge between v_i and v_j is $w_{i,j} \Leftarrow \infty (i, j = 1, \ldots, n)$ $(O(n^2))$ (line 2 in Algorithm 3).
- Step 2. $\forall v_i, v_j$ $(i \neq j, i, j = 1, \ldots, n)$, $w_{i,j} \Leftarrow 1$ if $V_{SPCL}^{[t_i,t_j]} \geq \epsilon_9$ $(O(n^3))$ (lines 3-12).
- Step 3. Find the shortest path from v_1 to v_n with the extension of the Dijkstra's algorithm ($O((n+m)\log n)$, where $\sum_{v_i} \deg(v_i) = 2m$, $\deg(v_i)$ is the degree of v_i) (lines 13-27).

Algorithm 3. Optimal MTI algorithm
Input: trust ratings $R^{(t_i)}$,
the given interval $[t_1, t_n]$,
the threshold ϵ_9 of V_{SPCL} (such as 0.9).
Output: the set of MTI with the minimal number of intervals in array t_b.

$t_{dimension} \Leftarrow t_n - t_1 +$ small time period unit
initialize the adjacent matrix $M \Leftarrow \infty$
for all i such that $1 \leq i \leq t_{dimension}$ **do**
  for all j such that $i \leq j \leq t_{dimension}$ **do**
    if $i == j$ **then**
      $M_{i,j} \Leftarrow 0$
    else if $V_{SPCL}^{[i,j]} \geq \epsilon_9$ **then**
      $M_{i,j} \Leftarrow 1$
      $M_{j,i} \Leftarrow 1$
    end if
  end for
end for
Let $path$ be shortest path from t_1, and $path \Leftarrow \emptyset$.
for all t_i in $[t_1, t_n]$ **do**
  let $l(t_i) = M_{t_1,t_i}$
end for
$t_b \Leftarrow \emptyset$.
while $\text{length}(path) < t_{dimension}$ **do**
  find t_i s.t. $l(t_i) = \min\{l(t_j) | t_j \in [t_1, t_n]$ and $t_j \in path\}$
  $path \Leftarrow path \cup t_i$
  for all $t_j \in path$ **do**
    if $l(t_i) + M_{t_i,t_j} < l(t_j)$ **then**
      $l(t_j) \Leftarrow l(t_i) + M_{t_i,t_j}$
      $t_b \Leftarrow t_b \cup t_i$
    end if
  end for
end while

Since n^3 dominates $m \log n$, the optimal MTI algorithm incurs the complexity of $O(n^3)$.

6 Experiments

In this section, we introduce the results of our experiments conducted on both real data sets and synthetic data sets. The aim of the experiments is to study the effectiveness and efficiency of our proposed vertical rating aggregation and horizontal rating aggregation approaches.

In these experiments, we set $\epsilon_1 = 0.06$ and $\epsilon_2 = 0.10$ which are the thresholds of relative rating density and marginal rating percentage, and set the service rating reputation threshold $\epsilon_3 = 0.5$. Meanwhile, we set $\epsilon_4 = 0.8$ and $\epsilon_5 = 0.0007$ which are the thresholds for determining the *coherent*, *upgoing*, *dropping*, or *uncertain* case of *STT*.

6.1 Experiment 1

In this experiment, we aim to illustrate why the service trust vector is necessary by comparing our method with two existing approaches in [29] and [33], because they are also based on nonbinary ratings only and apply to service-oriented environments. Both the ratings and corresponding regression lines are plotted in Fig. 6. The computed trust vectors are listed in Table 1.

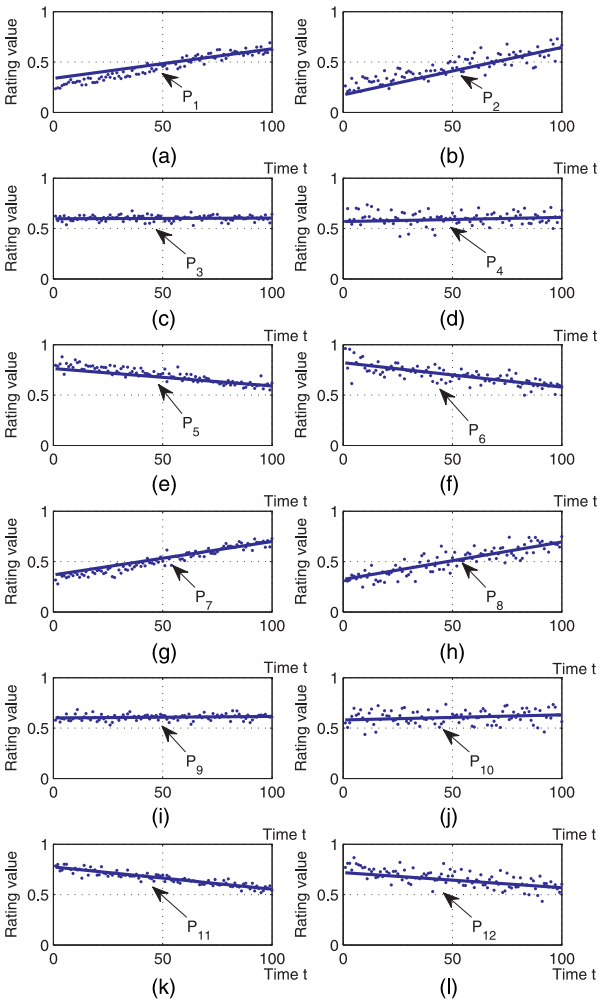

Fig. 6. Experiment 1.

In the comparison with the Sporas approach in [33], we evaluate the trust level of six service providers P_1 to P_6, with constant $\theta = 5$, acceleration factor $\sigma = 25$, the reputation of ratee $R_i^{other} = 1$, and initial reputation $R_0 = 0.1$. According to Table 1, all six service providers P_1 to P_6 (see Figs. 6a, 6b, 6c, 6d, 6e, and 6f) have almost the same V_{FTL}. Therefore, they seemingly have the same trust level. However, they have different V_{STT} or V_{SPCL}. Based on the properties introduced in Section 4.3, we can determine the trust trend as listed in Table 1, with which the six service providers P_1 to P_6 can be partially ordered: $P_1 > P_3 > P_5$, $P_2 > P_4 > P_6$, $P_1 > P_2$, $P_3 > P_4$, and $P_5 > P_6$.

Similarly, we compare our method with the approach in [29] for the trust evaluation of six service providers P_7 to P_{12} (see Figs. 6g, 6h, 6i, 6j, 6k, and 6l), with the scale control factor $\lambda = 1$, parameters $\alpha = 2$ and $\beta = 20$. From Table 1, we can notice that the six service providers P_7 to P_{12} have almost the same V_{FTL} but different V_{STT} or V_{SPCL}. Hence, the six service providers P_7 to P_{12} can be partially ordered: $P_7 > P_9 > P_{11}$, $P_8 > P_{10} > P_{12}$, $P_7 > P_8$, $P_9 > P_{10}$, and $P_{11} > P_{12}$.

From this experiment, we can observe that under some circumstances, a service trust vector can depict the trust level more precisely than a single trust value.

TABLE 1
Trust Vectors in Experiment 1

	V_{FTL}	V_{STT}	*STT*	V_{SPCL}
P_1	0.6048	0.0029	*upgoing*	0.9417
P_2	0.6034	0.0047	*upgoing*	0.8929
P_3	0.6036	0.0001	*coherent*	0.9573
P_4	0.6066	0.0004	*coherent*	0.9101
P_5	0.6055	-0.0017	*dropping*	0.9510
P_6	0.6004	-0.0024	*dropping*	0.9337
P_7	0.6084	0.0033	*upgoing*	0.9468
P_8	0.6010	0.0037	*upgoing*	0.9068
P_9	0.6031	0.0002	*coherent*	0.9618
P_{10}	0.6097	0.0005	*coherent*	0.8904
P_{11}	0.6034	-0.0022	*dropping*	0.9583
P_{12}	0.6055	-0.0015	*dropping*	0.9047

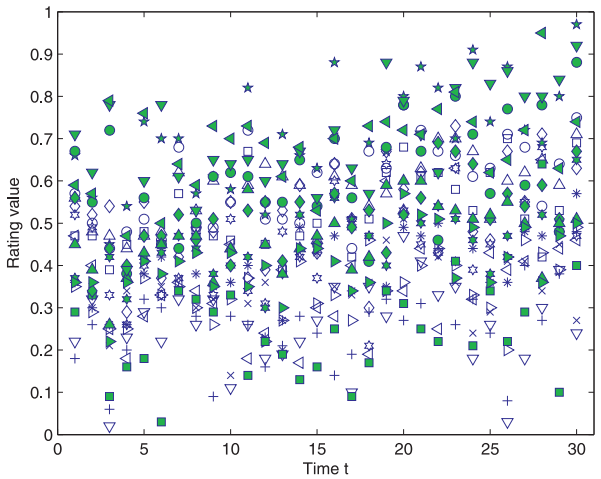

Fig. 7. Service rating values in Experiment 2.

6.2 Experiment 2

In this experiment, we introduce an example to illustrate our vertical rating aggregation approach. There are 20 service providers who obtain the trust ratings for the services delivered in the time interval $[t_1, t_{30}]$. The trust ratings $\{r_j^{(t_i)} | r_j^{(t_i)} \in [0,1]$ is for the service delivered at time t_i from provider $j\}$ for 20 service providers are plotted in Fig. 7.

- **Case 1.** In this case, we apply the vertical rating aggregation approach introduced in Section 3.1 without considering any service rating reputation (*SRR*). For each rating set, we first check if all the ratings conform to the Gaussian distribution hypothesis by applying the goodness-of-fit test procedure [9]. If the Gaussian distribution hypothesis holds, $R_c^{(t_i)}$, $R_u^{(t_i)}$, and $R_l^{(t_i)}$ are computed with (1), (2), and (3), respectively.

  If the Gaussian distribution hypothesis does not hold, then the clustering-based analysis method is applied. Taking the rating set at the small time period t_{10} (see Fig. 7) as an example, the corresponding histogram of ratings and frequency is plotted in Fig. 8. With the relative rating density threshold $\epsilon_1 = 0.06$, the centered cluster is $[0.25, 0.75]$, whose $P_{marginal}^{(t_{10})}$ is no more than the marginal rating

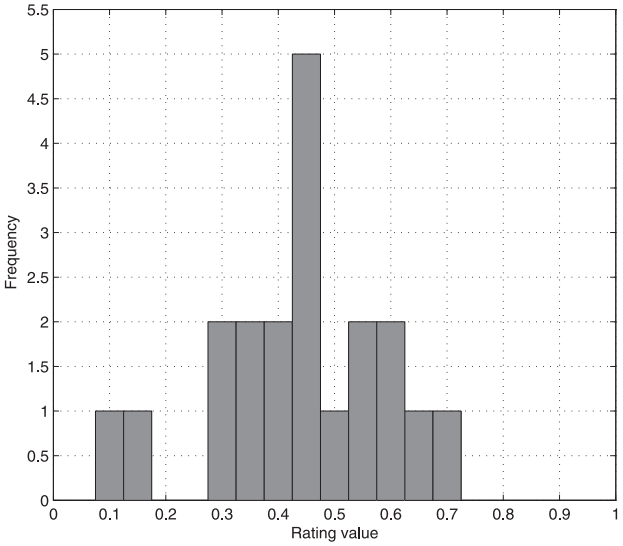

Fig. 8. Rating frequency at t_{10} in Experiment 2.

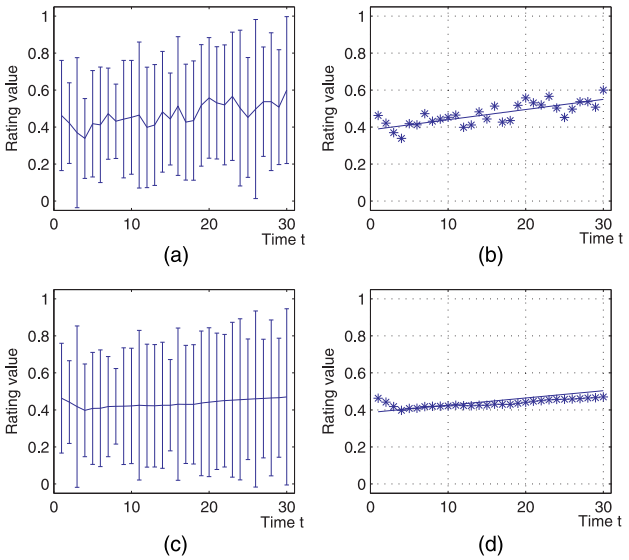

Fig. 9. Error bars and *STT* in Experiment 2.

TABLE 2
Trust Vectors in Experiment 2

$R^{(t_i)}$	V_{FTL}	V_{STT}	V_{SPCL}
Case 1 (without *SRR*)	0.4906	0.0055	0.9344
Case 2 (with *SRR*)	0.4617	0.0039	0.9697

percentage threshold $\epsilon_2 = 0.10$. Based on this cluster, we have $R^{(t_{10})} = 0.4526$. Thus, the ratings out of $[0.25, 0.75]$ are identified as marginal ratings.

In Fig. 9, there are 30 small time periods (i.e., for any t_i, $1 \leq i \leq 30$). For each rating set $\{r_j^{(t_i)}\}$, $R_c^{(t_i)}$, $R_u^{(t_i)}$, and $R_l^{(t_i)}$ are computed by either the Gaussian distribution-based analysis method or the clustering-based analysis method. These values are plotted by the error bars in Fig. 9a.

By applying the *STT* and *SPCL* evaluations, the trust vector is computed as listed in Table 2. The corresponding regression line is plotted in Fig. 9b together with 30 vertically aggregated ratings.

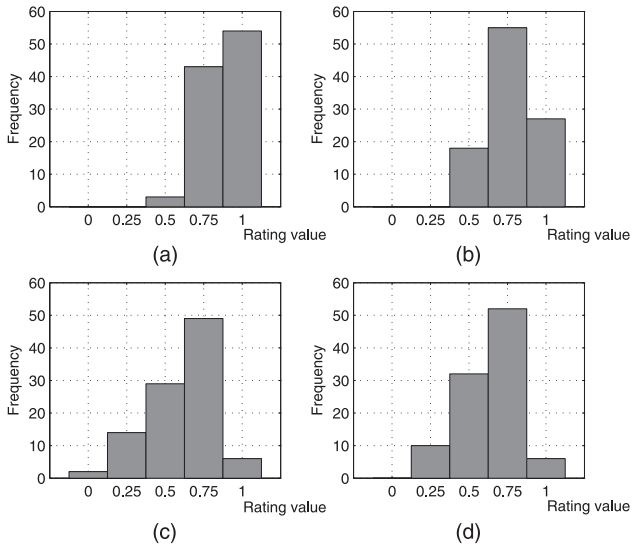

Fig. 10. Rating frequency examples in Experiment 3. (a) t_3 in Case 1. (b) t_6 in Case 2. (c) t_1 in Case 3. (d) t_1 in Case 4.

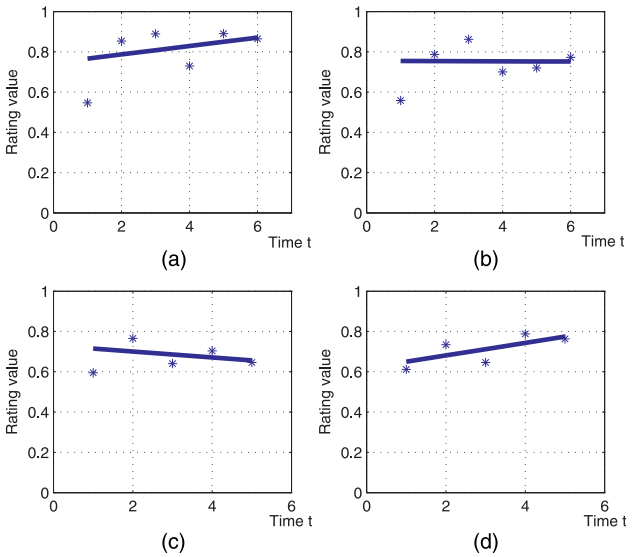

Fig. 11. Single trust vector in Experiment 3. (a) Case 1 $\varepsilon_9 = 0.8$. (b) Case 2 $\varepsilon_9 = 0.8$. (c) Case 3 $\varepsilon_9 = 0.8$. (d) Case 4 $\varepsilon_9 = 0.8$.

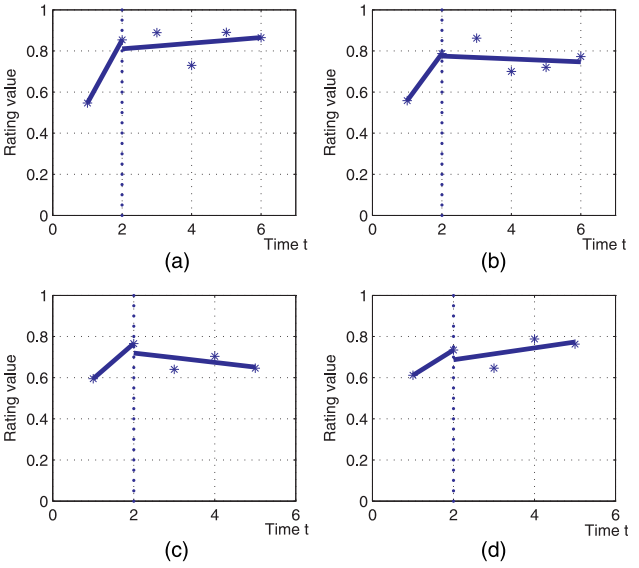

Fig. 12. MTI in Experiment 3. (a) Case 1 $\varepsilon_9 = 0.9$. (b) Case 2 $\varepsilon_9 = 0.91$. (c) Case 3 $\varepsilon_9 = 0.94$. (d) Case 4 $\varepsilon_9 = 0.923$.

- **Case 2.** In this case, we study the vertical aggregation with service rating reputation (*SRR*) using the threshold $\epsilon_2 = 0.10$. The computed $R_c^{(t_i)}$, $R_u^{(t_i)}$, and $R_l^{(t_i)}$ are plotted by the error bars in Fig. 9c.

  By applying the *STT* and *SPCL* evaluations with the service rating reputation threshold $\epsilon_3 = 0.5$, the trust vector is computed as listed in Table 2. The corresponding regression line is plotted in Fig. 9d together with 30 vertically aggregated ratings.

Compared with the curve of the vertically aggregated ratings $\{R^{(t_i)}\}$ in Case 1 as plotted in Fig. 9a, the curve of $\{R^{(t_i)}\}$ in Case 2 as plotted in Fig. 9c is smoother. This is because *SRR* is the accumulated rating reputation and it glues the ratings in two adjacent small time periods (i.e., t_i and t_{i+1}). Therefore, V_{SPCL} in Case 2 is greater than that in Case 1 (see Table 2) indicating that the trust vector in Case 2 can represent the ratings better.

6.3 Experiment 3

In this experiment, with the data of teaching evaluation and unit evaluation in up to six years collected at Macquarie University,[2] Sydney, Australia, we study both the vertical aggregation and the horizontal aggregation approaches.

At the end of each semester, the Center for Professional Development[3] at Macquarie University asks students to provide feedback on a teacher's teaching quality[4] and a unit's (a subject's) quality[5] using questionnaires.

In this experiment, we use two rating data sets of teaching quality (Cases 1 and 2 in Figs. 11 and 12) and two rating data sets of unit quality (Cases 3 and 4 in Figs. 11 and 12). Each data set of teaching quality consists of the ratings given to the same question in six years while each data set of unit quality is for five years.

We first study the vertical aggregation of the ratings given to a question in the same year. The data sets of four cases are plotted in Figs. 10a, 10b, 10c, and 10d, respectively. After vertical aggregation, the centered cluster is $[0.75, 1]$ in Case 1 (see Fig. 10a), $[0.5, 1]$ in Case 2 (see Fig. 10b), $[0.25, 0.75]$ in Case 3 (see Fig. 10c), and $[0.25, 0.75]$ in Case 4 (see Fig. 10d), respectively.

In Case 1 (see Fig. 10a), the vertically aggregated rating is 0.8892, which is larger than 0.8775—the average of ratings, because the rating 0.5 is taken as marginal and it is smaller than the ratings in the centered cluster. In Case 2 (see Fig. 10b), the vertically aggregated rating is 0.7725, which is the same as the average of ratings, because there is no marginal rating identified. In contrast, the vertically aggregated rating is 0.5951 in Case 3 (see Fig. 10c), which is smaller than the rating average 0.6075. This is because there are more marginal ratings 1 than marginal ratings 0. In Case 4 (see Fig. 10d), the vertically aggregated rating is 0.6117. It is smaller than the rating average 0.635 because 1

2. http://www.mq.edu.au.
3. http://www.cpd.mq.edu.au.
4. http://www.mq.edu.au/ltc/eval_teaching/teds.htm.
5. http://www.mq.edu.au/ltc/eval_teaching/leu.htm.

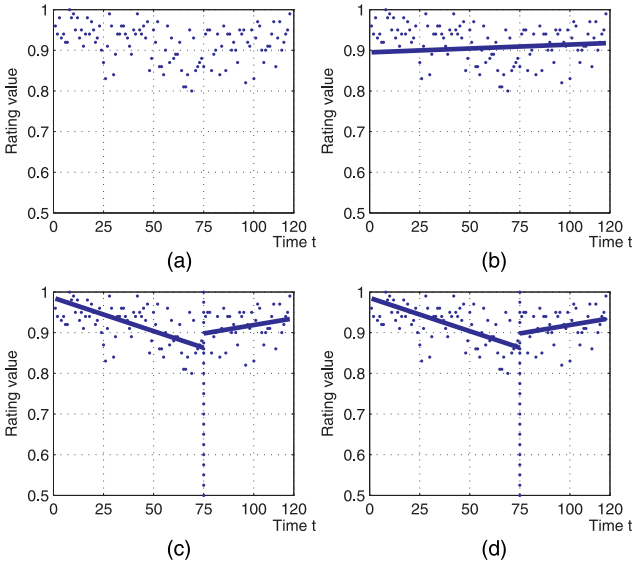

Fig. 13. MTI in Experiment 4. (a) Original data. (b) $\varepsilon_9 = 0.9$. (c) $\varepsilon_9 = 0.93$ (greedy). (d) $\varepsilon_9 = 0.93$ (optimal).

is the marginal rating and it is larger than the ratings in the centered cluster.

For horizontal aggregation, if we set the threshold $\epsilon_9 = 0.8$ for V_{SPCL}, as plotted in Fig. 11, one trust vector is obtained in each case. In contrast, a higher threshold ϵ_9 may lead to more trust vectors (i.e., more time intervals). As plotted in Fig. 12, two trust vectors are obtained if $\epsilon_9 = 0.9$ in Case 1 (see Fig. 12a), $\epsilon_9 = 0.91$ in Case 2 (see Fig. 12b), $\epsilon_9 = 0.94$ in Case 3 (see Fig. 12c), and $\epsilon_9 = 0.923$ in Case 4 (see Fig. 12d).

6.4 Experiment 4

In this experiment, we further study the horizontal aggregation approach with a large set of ratings from eBay [1].

The rating sample from an eBay seller consists of about 12,000 ratings in total about the transactions occurred in 118 days since February 2009. At eBay, a rating can be 1 ("positive"), 0 ("neutral"), or -1 ("negative"). Like the method used in [31], we calculate the *feedback score percentage* as $S_p = \frac{P-N}{P+Ne+N}$, where P, Ne, and N are the number of positive, neutral, and negative ratings, respectively. We use each day's ratings to compute the feedback score rate, which is taken as the rating $R^{(t_i)}$ and plotted in Fig. 13a. As a result, there are 118 time periods in total (i.e., $\forall t_i, 1 \leq i \leq 118$).

When $\epsilon_9 = 0.9$ for V_{SPCL}, as plotted in Fig. 13b, with either the greedy algorithm or the optimal algorithm, only one trust vector is obtained covering t_1 to t_{118}. With a higher threshold $\epsilon_9 = 0.93$, two time intervals are obtained with either the greedy algorithm (see Fig. 13c) or the optimal algorithm (see Fig. 13d).

We also notice that, when $\epsilon_9 = 0.94$, the greedy algorithm outputs 11 time intervals (see Fig. 14a). In contrast, the optimal algorithm outputs eight time intervals only (see Fig. 14b). Thus, the optimal algorithm may output a smaller set of MTI.

6.5 Experiment 5

In this experiment, we use large-scale synthetic rating data sets to compare the efficiency of our proposed greedy MTI algorithm and the optimal MTI algorithm.

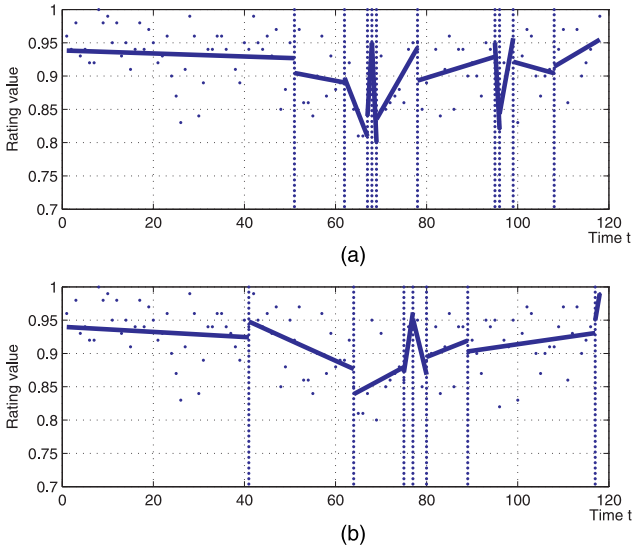

Fig. 14. Comparison of greedy and optimal algorithms in Experiment 4. (a) $\varepsilon_9 = 0.94$ (greedy). (b) $\varepsilon_9 = 0.94$ (optimal).

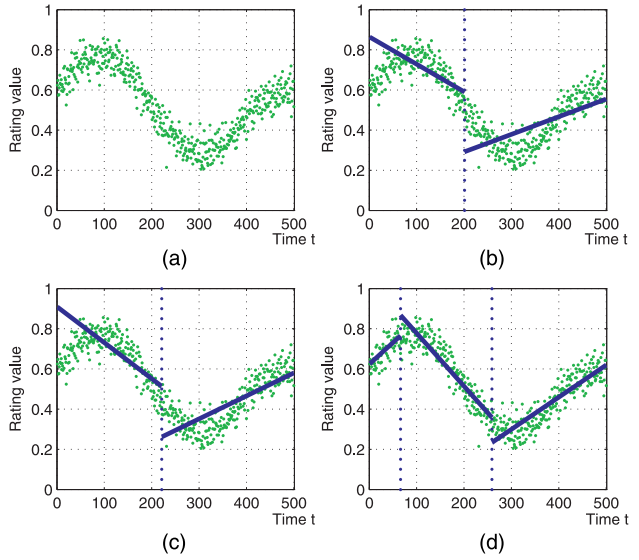

Fig. 15. Case 1 in Experiment 5. (a) Vertically aggregated ratings. (b) $\varepsilon_9 = 0.85$. (c) $\varepsilon_9 = 0.87$. (d) $\varepsilon_9 = 0.9$.

We conducted our experiments on top of Matlab 7.6.0.324 (R2008a) running on a Dell Vostro V1310 laptop with an Intel Core 2 Duo T5870 2.00 GHz CPU and 3 GB RAM. Each result of the CPU time is the average of three independent executions.

In this experiment, we use three sets of ratings. Each set consists of 50,000 trust ratings distributed in 500 time periods. In each time period t_i, there are 100 trust ratings, which are vertically aggregated yielding one rating $R^{(t_i)}$. All computed $\{R^{(t_i)}\}$ for three cases are plotted in Figs. 15a, 16a, and 17a, respectively.

By applying the optimal algorithm to Case 1 (see Fig. 15a), if the threshold is set as $\epsilon_9 = 0.85$, 0.87, or 0.9, we can obtain two or three time intervals as plotted in Figs. 15b, 15c, and 15d, respectively. As the trust trend changes more frequently in Case 2 (see Fig. 16a), when the threshold is set as $\epsilon_9 = 0.85$, 0.87, or 0.9, six to eight time intervals are obtained (see Figs. 16b, 16c, and 16d). In contrast, in Case 3

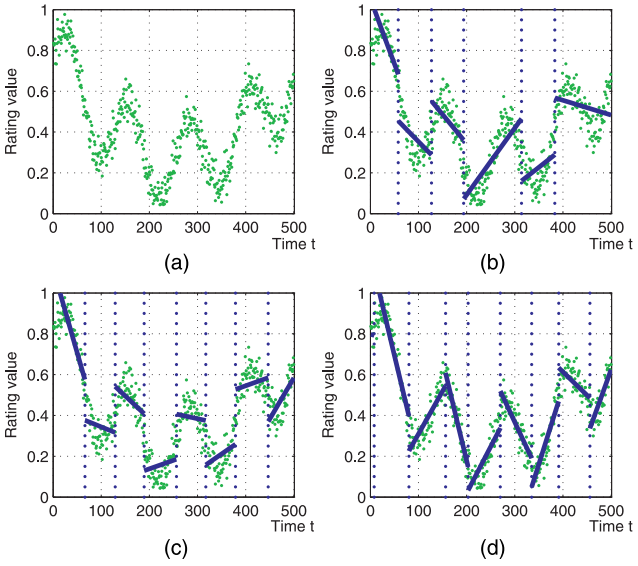

Fig. 16. Case 2 in Experiment 5. (a) Vertically aggregated ratings. (b) $\varepsilon_9 = 0.85$. (c) $\varepsilon_9 = 0.87$. (d) $\varepsilon_9 = 0.9$.

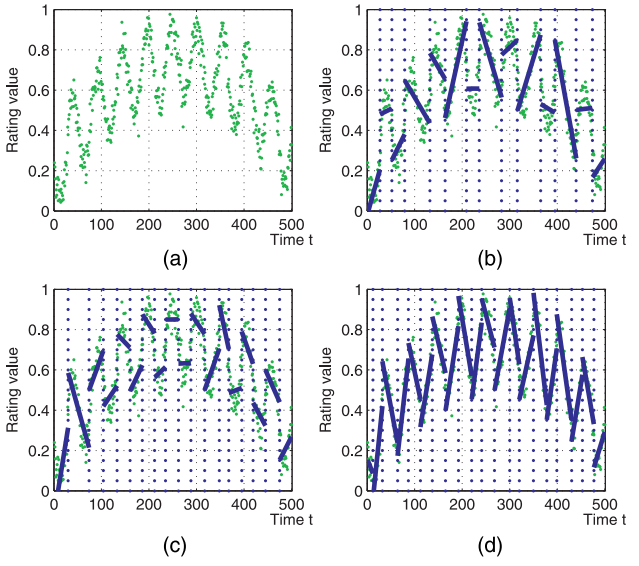

Fig. 17. Case 3 in Experiment 5. (a) Vertically aggregated ratings. (b) $\varepsilon_9 = 0.85$. (c) $\varepsilon_9 = 0.87$. (d) $\varepsilon_9 = 0.9$.

(see Fig. 17a), the trust trend changes the most frequently in all three cases. With the same thresholds $\epsilon_9 = 0.85$, 0.87, or 0.9, there are 14 (see Fig. 17b), 18 (see Fig. 17c), or 20 time intervals (see Fig. 17d) obtained. Thus, in three cases, we can use 3, 8, or 20 trust vectors to approximately represent 50,000 trust ratings. Namely, a small set of values can represent a large set of trust ratings with the trust features well retained.

In addition, in three cases with different thresholds, we compare the CPU time of the greedy algorithm with that of the optimal algorithm. From the results listed in Table 3, We can observe the following results:

1. For trust rating aggregation, generally the greedy algorithm runs faster than the optimal algorithm though they have the same time complexity.
2. In the optimal algorithm, the generation of the adjacency matrix (see Steps 1 and 2 in the optimal algorithm introduced in Section 5.2 and "Steps 1&2" in Table 3) takes most time as it should check if $V_{SPCL} \geq \epsilon_9$ for all $\frac{n^2}{2} - n$ possible edges. In contrast, the Dijkstra's algorithm-based MTI analysis takes a little proportion of time as listed in "Step 3 of the optimal algorithm" in Table 3.
3. We can observe that from Case 1 to Case 3 (see Figs. 15a, 16a, and 17a), the trust trend changes more frequently. This leads to more CPU time for the greedy algorithm to find MTI with the same threshold. However, the CPU time of the optimal algorithm is almost unchanged.
4. When the threshold ϵ_9 becomes larger, the CPU time of the greedy algorithm becomes longer while it changes very little in the optimal algorithm. When ϵ_9 approaches 1, both algorithms consume similar CPU time.

Thus, incorporating the results in both Experiments 4 and 5, the greedy algorithm is useful and more efficient when processing rating data with less frequently changing trust trend. In contrast, the optimal algorithm can outperform when processing large-scale rating data with a high V_{SPCL} threshold.

TABLE 3
CPU Time in Seconds in Experiment 5

	Time interval	ϵ_9	Optimal algorithm			Greedy algorithm
			Steps 1&2	Step 3	Total	
Case 1	[1,500]	0.85	1344.6	1.5	1346.4	3.6
Case 1	[1,500]	0.87	1336.9	1.5	1338.7	4.4
Case 1	[1,500]	0.9	1337.4	1.5	1339.2	9.7
Case 1	[1,500]	1	1269.1	1.5	1271.0	1290.8
Case 2	[1,500]	0.85	1300.0	1.4	1301.7	15.6
Case 2	[1,500]	0.87	1294.1	1.4	1295.8	18.4
Case 2	[1,500]	0.9	1295.1	1.4	1295.8	25.7
Case 2	[1,500]	1	1265.3	1.5	1267.2	1299.1
Case 3	[1,500]	0.85	1308.7	1.4	1310.6	35.9
Case 3	[1,500]	0.87	1301.0	1.4	1302.7	46.3
Case 3	[1,500]	0.9	1298.8	1.4	1300.6	54.4
Case 3	[1,500]	1	1268.7	1.5	1270.6	1292.4

7 Conclusions

In this paper, we have proposed a novel two-dimensional aggregation approach that aggregates a large set of trust ratings both vertically and horizontally. In the vertical aggregation of trust ratings, we adopt the Gaussian distribution-based analysis method and the clustering-based analysis method. For the horizontal aggregation of trust ratings, we propose the service trust vector approach and the multiple time interval (MTI) analysis approach including the greedy MTI algorithm and the optimal MTI algorithm. The trust vector provides more meaningful information that can be used for service provider comparison and selection. The proposed optimal MTI analysis algorithm can ensure the representation of a large set of trust ratings with the minimal number of values while highly preserving the trust features. Therefore, our work is significant for large-scale trust data management, transmission, and evaluation.

In the optimal MTI algorithm, given a set of ratings and the same threshold, several minimal sets of MTI may exist. Thus, in our future work, the optimal algorithm can be further extended to find the best MTI set with the largest summation of the *SPCL* values.

TABLE 4
Notations Used in This Paper

Notation	Representation	First occurrence
a_0; a_1	regression line $R = a_0 + a_1 t$	Section 4.2
$D_r^{(t_i)}$	relative rating density	Section 3.1.2
$fre(r_j^{(t_i)})$	the frequency of $r_j^{(t_i)}$ delivered at t_i	Section 3.1.2
FTL	final trust level	Section 1
m	the number of ratings for the services delivered at t_i	Section 3
m_0	argument to control the function curve	Definition 4
m'	the number of ratings in $[R_l^{(t_i)}, R_u^{(t_i)}]$	Definition 2
m''	the number of trust ratings in $[R_l^{(t_i)'}, R_u^{(t_i)'}]$	Definition 6
MTI	multiple time interval	Section 5
n_r	the size of the dataset	Section 3.1.2
$n_{marginal}^{(t_i)}$	the number of marginal ratings delivered at t_i	Section 3.1.2
$n_{total}^{(t_i)}$	the total number of ratings delivered at t_i	Section 3.1.2
$P_{marginal}^{(t_i)}$	the marginal rating percentage at t_i	Section 3.1.2
$r_j^{(t_i)}$	rating from client j for the services delivered at t_i	Definition 1
$r_{j_{dis}}^{(t_i)}$	the distance between $r_j^{(t_i)}$ and $R_c^{(t_i)'}$	Principle 2
$r_k^{(t_i)'}$	rating in the range of $[R_l^{(t_i)}, R_u^{(t_i)}]$	Definition 2
$r_k^{(t_i)''}$	trust rating in the range of $[R_l^{(t_i)'}, R_u^{(t_i)'}]$	Definition 6
$\bar{\mathbf{r}}^{(t_i)}$	the rating that would ideally represent the trust level of the service delivered at t_i	Section 3.1
$R^{(t_i)}$	vertically aggregated rating at t_i	Definition 2
$R^{(t_i)'}$	weighted vertically aggregated rating	Section 3.2.2
$R_c^{(t_i)}$	centerline of ratings $\{R_c^{(t_i)}\}$	Definition 1
$R_c^{(t_i)'}$	weighted centerline	Definition 5
$R_l^{(t_i)}$	lower control limit	Section 3.1
$R_l^{(t_i)'}$	weighted lower control limit	Definition 5
$R_{SRR_j}^{(t_i)}$	the *SRR* value for client j at t_i	Principle 2
$R_u^{(t_i)}$	upper control limit	Section 3.1
$R_u^{(t_i)'}$	weighted upper control limit	Definition 5
S	sum of squares of the distance d_i	Eq. (16)
$SPCL$	service performance consistency level	Section 1
SRR	service rating reputation	Section 3.2
STT	service trust trend	Section 1
t_i	a small time period	Section 3
$\overline{T}^{[t_{start}, t_{end}]}$	service trust vector	Definition 9
$v_{t_{dis}}^{[t_{start}, t_{end}]}$	weighted average distance	Definition 7
$V_{FTL}^{[t_{start}, t_{end}]}$	the *FTL* value for time interval $[t_{start}, t_{end}]$	Eq. (15)
$V_{SPCL}^{[t_{start}, t_{end}]}$	the *SPCL* value for time interval $[t_{start}, t_{end}]$	Definition 8
$V_{SRR_j}^{(t_i)}$	the *SRR* value for client j from t_1 to t_i	Definition 3
$V_{STT_j}^{[t_{start}, t_{end}]}$	the *STT* value for time interval $[t_{start}, t_{end}]$	Definition 9
w_{t_i}	weight at t_i	Definition 3
γ	$\max r_{j_{dis}}^{(t_i)}$	Definition 4
ϵ_1	the threshold of relative rating density	Section 6
ϵ_2	the threshold of marginal rating percentage	Section 6
ϵ_3	the threshold of $R_{SRR_k}^{(t_i)}$	Definition 6
ϵ_4; ϵ_5	thresholds used to determine the case of *STT*	Section 4.3
ϵ_6; ϵ_7; ϵ_8	thresholds used to compare trust vector	Section 4.4
ϵ_9	the threshold of V_{SPCL} used to determine MTI	Section 5

APPENDIX

Notations are given in Table 4.

REFERENCES

[1] eBay, http://www.eBay.com, 2011.

[2] GNutella, http://www.gnutella.com, 2009.

[3] World Wide Web Consortium (W3C), http://www.w3.org, 2011.

[4] E. Damiani, S.D.C. di Vimercati, S. Paraboschi, P. Samarati, and F. Violante, "A Reputation-Based Approach for Choosing Reliable Resources in Peer-to-Peer Networks," *Proc. ACM Conf. Computer and Comm. Security (CCS '02)*, pp. 207-216, 2002.

[5] E. Damiani, S.D.C. di Vimercati, P. Samarati, and M. Viviani, "A Wowa-Based Aggregation Technique on Trust Values Connected to Metadata," *Electronic Notes in Theoretical Computer Science*, vol. 157, no. 3, pp. 131-142, 2006.

[6] E.W. Dijkstra, "A Note on Two Problems in Connexion with Graphs," *Numerische Mathematik*, vol. 1, pp. 269-271, 1959.

[7] N. Griffiths, "Task Delegation Using Experience-Based Multi-Dimensional Trust," *Proc. Int'l Joint Conf. Autonomous Agents and Multiagent Systems (AAMAS '05)*, pp. 489-496, 2005.

[8] J. Han and M. Kamber, *Data Mining: Concepts and Techniques*. Morgan Kaufmann, 2000.

[9] W.W. Hines, D.C. Montgomery, D.M. Goldsman, and C.M. Borror, *Probability and Statistics in Engineering*. John Wiley & Sons, 2003.

[10] N. Hu, P.A. Pavlou, and J. Zhang, "Can Online Reviews Reveal a Product's True Quality?: Empirical Findings and Analytical Modeling of Online Word-of-Mouth Communication," *Proc. ACM Conf. Electronic Commerce (EC '06)*, pp. 324-330, 2006.

[11] T.D. Huynh, N.R. Jennings, and N.R. Shadbolt, "An Integrated Trust and Reputation Model for Open Multi-Agent Systems," *Autonomous Agents and Multi-Agent Systems*, vol. 13, no. 2, pp. 119-154, 2006.

[12] A. Jøsang, "Subjective Evidential Reasoning," *Proc. Int'l Conf. Information Processing and Management of Uncertainty (IPMU '02)*, 2002.

[13] A. Jøsang, R. Ismail, and C. Boyd, "A Survey of Trust and Reputation Systems for Online Service Provision," *Decision Support Systems*, vol. 43, no. 2, pp. 618-644, 2007.

[14] R. Jurca and B. Faltings, "Minimum Payments that Reward Honest Reputation Feedback," *Proc. ACM Conf. Electronic Commerce (EC '06)*, pp. 190-199, 2006.

[15] R. Jurca and B. Faltings, "Collusion-Resistant, Incentive-Compatible Feedback Payments," *Proc. ACM Conf. Electronic Commerce (EC '07)*, pp. 200-209, 2007.

[16] S.D. Kamvar, M.T. Schlosser, and H. Garcia-Molina, "The Eigentrust Algorithm for Reputation Management in P2P Networks," *Proc. Int'l Conf. World Wide Web (WWW '03)*, pp. 640-651, 2003.

[17] D.H. Knight and N.L. Chervany, "The Meaning of Trust," Technical Report WP9604, Management Information Systems Research Center, Univ. of Minnesota, 1996.

[18] R.F. Korte, "Biases in Decision Making and Implications for Human Resource Development," *Advances in Developing Human Resources*, vol. 5, no. 4, pp. 440-457, 2003.

[19] L. Li and Y. Wang, "A Trust Vector Approach to Service-Oriented Applications," *Proc. Int'l Conf. Web Services (ICWS '08)*, pp. 270-277, 2008.

[20] L. Li and Y. Wang, "Subjective Trust Inference in Composite Services," *Proc. AAAI Conf. Artificial Intelligence (AAAI '10)*, July 2010.

[21] L. Li, Y. Wang, and V. Varadharajan, "Fuzzy Regression Based Trust Prediction in Service-Oriented Applications," *Proc. Int'l Conf. Autonomic and Trusted Computing (ATC '09)*, pp. 221-235, 2009.

[22] K.-J. Lin, H. Lu, T. Yu, and C. en Tai, "A Reputation and Trust Management Broker Framework for Web Applications," *Proc. IEEE Int'l Conf. e-Technology, e-Commerce and e-Service (EEE '05)*, pp. 262-269, 2005.

[23] M.P. Papazoglou, "Web Services and Business Transactions," *World Wide Web*, vol. 6, no. 1, pp. 49-91, 2003.

[24] I. Ray and S. Chakraborty, "A Vector Model of Trust for Developing Trustworthy Systems," *Proc. Ninth European Symp. Research Computer Security*, pp. 260-275, 2004.

[25] W.T.L. Teacy, J. Patel, N.R. Jennings, and M. Luck, "Travos: Trust and Reputation in the Context of Inaccurate Information Sources," *Autonomous Agents and Multi-Agent Systems*, vol. 12, no. 2, pp. 183-198, 2006.

[26] L.-H. Vu, M. Hauswirth, and K. Aberer, "QoS-Based Service Selection and Ranking with Trust and Reputation Management," *Proc. Cooperative Information System Conf. (CoopIS '05)*, pp. 466-483, 2005.

[27] Y. Wang and E.-P. Lim, "The Evaluation of Situational Transaction Trust in E-Service Environments," *Proc. IEEE Int'l Conf. e-Business Eng. (ICEBE '08)*, pp. 265-272, 2008.

[28] Y. Wang and K.-J. Lin, "Reputation-Oriented Trustworthy Computing in E-Commerce Environments," *IEEE Internet Computing*, vol. 12, no. 4, pp. 55-59, July/Aug. 2008.

[29] Y. Wang, K.-J. Lin, D.S. Wong, and V. Varadharajan, "Trust Management towards Service-Oriented Applications," *Service Oriented Computing and Applications*, vol. 3, no. 2, pp. 129-146, 2009.

[30] Y. Wang and V. Varadharajan, "Interaction Trust Evaluation in Decentralized Environments," *Proc. Int'l Conf. E-Commerce and Web Technologies (EC-Web '04),* pp. 144-153, 2004.
[31] L. Xiong and L. Liu, "PeerTrust: Supporting Reputation-Based Trust for Peer-to-Peer Electronic Communities," *IEEE Trans. Knowledge and Data Eng.,* vol. 16, no. 7, pp. 843-857, July 2004.
[32] B. Yu and M.P. Singh, "An Evidential Model of Distributed Reputation Management," *Proc. Int'l Joint Conf. Autonomous Agents and Multiagent Systems (AAMAS '02),* pp. 294-301, 2002.
[33] G. Zacharia and P. Maes, "Trust Management through Reputation Mechanisms," *Applied Artificial Intelligence,* vol. 14, no. 9, pp. 881-907, 2000.
[34] H. Zhao and X. Li, "Vectortrust: Trust Vector Aggregation Scheme for Trust Management in Peer-to-Peer Networks," *Proc. 18th Int'l Conf. Computer Comm. and Networks,* pp. 1-6, 2009.

Yan Wang received the BEng, MEng, and DEng degrees in computer science and technology in 1988, 1991, and 1996, respectively, from the Harbin Institute of Technology (HIT), P.R. China. He is now an associate professor in the Department of Computing, Macquarie University, Australia. Prior to joining Macquarie University, he was a research fellow in the Department of Computing Science, National University of Singapore, from 1999 to 2003. He has served as a PC member of more than 30 international conferences. He is an editor of the *International Journal of Web Engineering and Technology* (IJWET) and *Service-Oriented Computing & Applications* (SOCA), and was a guest coeditor of the special track on e-commerce of *IEEE Internet Computing* in 2008. His research interests include trust computing, e-commerce, software agent, and security. He is a senior member of the IEEE.

Lei Li received the BSc degree in information and computational science from Jilin University, P.R. China, in 2004, and the MSc degree in applied mathematics from the Memorial University of Newfoundland, Canada, in 2006. He is working toward the PhD degree in computer science at Macquarie University, Australia. His current research focuses on service-oriented reputation and trust management, and trust-oriented service selection and service composition. He is a student member of the IEEE.