# Bi-directional Trust Index Computation in Resource Marketplace

Avinash Sharma
Xerox Research Centre India
Bangalore, India

Tridib Mukherjee
Xerox Research Centre India
Bangalore, India

Partha Dutta
Xerox Research Centre India
Bangalore, India

Vinay Hegde
IIIT-B
Bangalore, India

## ABSTRACT

Commoditizing idle computing resources by sharing them in a marketplace has gained increased attention in recent years as a potential disruption to the modern cloud-based service delivery. Recent initiatives have focused on scavenging for idle resources and provide suitable incentives accordingly. A recent work on resources marketplace has proposed a Marketplace for Compute Infrastructure that not only allows resource owners to get incentives by sharing resources to the marketplace but also ensures Service Level Agreements (SLAs), such as performance guarantees, for the computing jobs to be run by the shared resources.

This paper proposes a Trust for Resource Marketplace (TRM) system that computes the trust level among the entities in a resource marketplace (RM), by incorporating key aspects of the interactions among these entities. In particular, an RM has three kinds of entities: users (with task requests), resources (on which task is executed), and resource owners. Over these entities, the system allows two kinds of trust queries: (i) for a user, a trust indexing of resources or resource owners and (ii) for a resource owner, a trust indexing of users. This is achieved by a novel interaction graph modelling followed by spectral analysis of this graph, thereby, capturing both direct and indirect relationships among RM entities while deriving trust indexes. Experiments with a combination of real and synthesized traces on TRM implementation show that the proposed trust computation can capture indirect relationship among entities and is robust against limited changes in topology.

## Keywords

Trust, Resource Marketplace, Spectral Graph Analysis

## 1. INTRODUCTION

Marketplaces for sharing idle computing resources have recently been proposed as a potential disruption to the modern cloud-based service delivery [1, 5]. Also, Bring Your Own Device (BYOD) is getting increased attention, especially in the growth markets. A marketplace for sharing idle resources can enable an organization to provide cloud-like infrastructure experience to users/employees out of their own devices [2]. Traditionally, volunteer computing have allowed resources to be shared for execution of computing tasks in voluntary manner (e.g., without any monetary incentive to the resource owner) [6]. Some recent initiatives have focused on scavenging for idle resources and provide proper incentives accordingly [5, 7]. [1] has proposed a Marketplace for Compute Infrastructure that not only allows resource owners to get incentives by sharing resources to the marketplace but also ensures Service Level Agreements (SLAs), such as performance guarantees, for the computing jobs to be run by the shared resources.

Such Resource Marketplaces (RMs) have three kinds of entities: users (entities requesting task to be executed), resources (entities on which tasks are executed) and resource owners. A major concern for a resource marketplace is the trust among these entities, since the users and resource owner may be unknown to each other. This paper outlines a Trust for Resource Marketplace (TRM) system that computes a trust index among entities in a resource marketplace.

A trust index can be useful for the RM entities in multiple ways. First, the users would want to have certain level of reliability or trustworthiness from the resource, in terms of whether their tasks execute successfully with the specified performance guarantees. Reliability of a resource can be gauged by a trust index that takes into account the resources' historical behaviour (i.e. whether a resource had successfully executed a certain types of request with performance guarantees) and resource owner's relationship to the user (e.g., if they belong to the same or related organizations or belong to same social community). A trust index is also helpful for a user when the user wants to maintain a level of security and privacy for her task or the associated data, which is typically the case in many modern applications. The security requirement can also persist in case the users do not want to share any proprietary process or algorithm to some outside party.

On the other hand, a resource owner may want a certain level of reliability from the marketplace on the type of applications that execute on her resources. The owner may want to avoid an application that overloads the resources (which


*SAC'15* April 13-17, 2015, Salamanca, Spain.

http://dx.doi.org/10.1145/2695664.2695792

may be detrimental to the resources' hardware health) or an application that is malicious (e.g., contains some computer malware).

## 1.1 Our Contribution

The contributions of this work are three fold. *The primary contribution* is the introduction of notion of trust among different entities in a resource marketplace (RM). The bi-directional trust index introduced in this work enables, 1)User specific trust ranking of resources, as well as 2) Resource owner specific trust ranking of users.

*Secondly*, we propose the novel interaction graph modelling for RM which enables considering multiple factors, such as historical usage patterns, organizational relationships as well as inter and intra-group social network interactions, while computing trust. Additionally, the spectral analysis based trust computation exploits both the direct and indirect interaction among RM entities modelled in the interaction graph.

*Thirdly*, we have implemented the proposed trust computation methods and evaluated them with real and synthesized traces. Experimental results show that the trust computation can capture indirect relationship while computing trust among entities and it robust against limited changes in number and weight of their connection edges.

# 2. RELATED WORKS

The notion of trust has been studied in multiple distributed computing setup such as Volunteer Computing (VC) and Peer-to-Peer (P2P) computing. In VC [6, 9], resource owners donate their computing resources to a specific project. However, as there is no monetary payment for using resources, there are no or limited SLAs or performance guarantees. In P2P setting, [8] outlines an approach in which peers make use of trust values to determine from which peer to download files, where a unique global trust value is computed for each peer based on the peers' history of uploads.

Trust has also been studied in the context of social networks. [10] leverages pre-established trust formed through friend relationships in a social network, which is used to share resources among the users. [13] use the distance in a social network for computing trust levels. [11] deals with predefined relationships of trust amongst users in a system that enables the sharing of resources between these users. Finally, [12] defines computing environments of different enterprises, which interact within a federated computing environment and rely upon a trust service to manage the interaction between the federation partners.

Recently, a framework for evaluating trust of service providers in cloud marketplaces was introduced in [3, 4]. The proposed technique enables verifying the capabilities captured in the Cloud Security Alliance's framework and also provides a decision model that checks consumer requirements against the verification results. The underlying verification uses hard trust based on rigid validation along with soft trust based on evidence about past behaviour.

While the above prior work consider the notion of trust in various distributed environments, none have been proposed for a resource marketplace (RM). Trust computing in a resource marketplace requires a composition of historical usage patterns (including task SLAs, performance measures and user ratings), organizational relationships, as well as, inter and intra-group social network interactions among users and resource owners. Also, different from earlier work, trust computation in this work considers direct and indirect interactions between the entities in a marketplace, by computing average connectivity between entities in the interaction graph of the RM. Finally, none of the earlier work computes bi-directional trust while sharing resources: users would prefer trusted resources, and resource owners would prefer tasks from trusted users.

# 3. SYSTEM DESCRIPTION

This paper outlines a Trust for Resource Marketplace (TRM) system (see Figure 1) that computes the trust level among the entities in an RM, by incorporating various aspects of the interactions among these entities. In particular, an RM has three kinds of entities: users (with task requests), resources (on which task is executed), and resource owners. Over these entities, the system allows two kinds of trust queries: (i) for a user, a trust indexing of resources or resource owners (ii) for a resource owner, a trust indexing of users.

The TRM system is composed of a Trust Computation Engine (TCE), which performs the trust computation, and multiple Trust Databases (TDB), which stores the profiles and historical usage information. In the current setup, the TRM interacts (e.g., serves trust related queries) with the RM. However, the proposed trust computation method applies to other configurations where TRM directly interacts with the users, resources and resource owners.

The trust computation in TCE is derived from the interaction graph between the entities in RM, where the nodes are users, resources or resource owners, and the edges (and their weights) capture relationships or interactions among the entities. This interaction graph is created by a graph induction (GI) module. GI uses different weighing functions to assign edge weights to the interaction graph. These weighing functions consider historical usage patterns in RM, as well as geographical, organizations or social network relationships among users and resource owners.

Next, the interaction graph created by GI is used by a Spectral Analysis (SA) module for computing trust indices. SA performs trust computation using spectral graph analysis of the interaction graph. In particular, SA computes a spectral (or Laplacian) embedding of the interaction graph that preserves the local neighbourhood structure and connectivity of the interaction graph. Finally, given an entity or node in RM, trust indices for other entities are computed by finding and ranking the Euclidean distances between the respective graph nodes in the embedded space. Computing trust between a pair of entities using their distance in the Laplacian embedding space has the advantage that it simultaneously considers all possible paths (or connections) between those entities in the original interaction graph, and it is scalable to large number of trust queries: once the embedding is computed for the interaction graph, it can be used to quickly compute trust level between any pair of entities.

## 3.1 Graph Induction

Graph induction module performs a domain specific modelling of the interaction graph of RM that captures the information relevant for trust computation. The interaction graph is modelled as a weighted undirected graph where each pair of nodes (vertices) can have at most one undirected positively weighted edge connecting them. The graph is defined

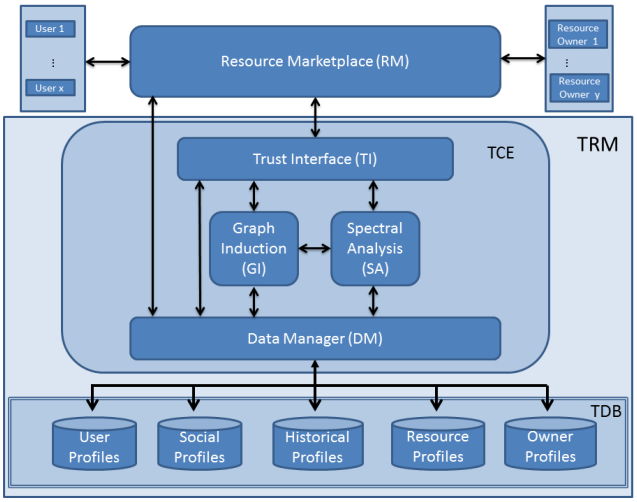



Figure 1: Trust for Resource Marketplace Architecture.

as follows:

- Vertex definition: Each entity of RM is modelled as unique vertex of the interaction graph. There are three type of entities or vertices, namely, Users, Resources and Owners.

- Edge definition: The edges model the relationship among RM entities. There are six type of edges in the interaction graph. The first three type of edges model connectivity/interaction between same type of vertices, i.e., User-User, Resource-Resource and Owner-Owner edges. These edges capture the relationship between same type of RM entity based on their profile attributes, e.g., users belonging to same organization or social group. The remaining three type of edges captures User-Resource, Resource-Owner and Owner-User relationship. The latter two types are binary relationship in the sense that they model resource ownership and scenarios where a user also owns some resource(s). The User-Resource edges encode historical usage patterns in the RM. These User-Resource edges are important for trust computation, as they allow the trust computation method to induce a trust definition based on historical experience of a user (or resource owner).

| | |
|---|---|
| User-User | $f_{uu}(social, location, organization)$ |
| Owner-Owner | $f_{oo}(social, location, organization)$ |
| Resource-Resource | $frr(location, capability)$ |
| Owner-Resource | whether owns a resource; constant value |
| User-Resource | $f_{ur}(usage\_freq, avg\_feedback)$ |
| User-Owner | whether same entity; constant value |

Table 1: List of functions used for computing edge weighs.

- Edge weight functions: The edge weights always have positive valued, normalized between 0 and 1, where 0 denotes no direct connectivity, and 1 denotes very strong connectivity relationship (between the edge endpoints). Although these edge weights reflect direct relationship between entities, the overall graph structure (topology information) captures the indirect interactions that are also important for defining the trust index. Table 1 summarizes list of weighting functions with associated parameters.

There can be multiple preferences while computing edge weights in the graph representation, e.g., some systems might want to give lower importance to historical usage patterns and higher preference to a resource belonging to same organization/geographical location, or vice-versa. Therefore, different entity attributes act as parameters to weight functions where these attributes along with their relative importance factor is used to generate a positive real value as edge weight. For example, fuu function (that is used to compute user-user edge weights) has three input parameter attributes. So for a given pair of users, a positive value between 0 and 1 is first assigned to each attribute based on their similarity (e.g. affiliation to same organization) and later a weighted linear combination of these values is computed along with respective importance factor in order to compute a real positive weight value. These importance factors are explicitly provided the GI module from RM via the TI module. As these preferences can change over time, the weights can be recomputed in fixed temporal cycles or on specific request from RM.

Using these definitions, an interaction graph model for TRM is induced with the following method.

**INPUT:** RM entities, TDB, attribute importance factors

**OUTPUT:** A TRM interaction graph

I For each RM entity instantiate a graph vertex.

II For each pair of entities belonging to same class (e.g. user-user, resource-resource, or owner-owner), fetch entity attributes from TDB and feed these attributes to edge weighing functions (e.g. $f_{uu}$) along with attribute importance factors, to compute edge weight value.

III For each pair of entities belonging to user-owner and owner-resource, specify a pre-determined fixed edge weight if relationships user 'is also a' owner and owner 'owns' resource hold.

IV For each pair of user-resource entities, fetch historical performance data from TDB and along with attribute importance factor, and feed these information to the edge weighting function fur to compute the edge weights.

The interaction graph obtained by above method is used to derive trust relationships among RM entities (detailed explanation is provided while describing Spectral Analysis module and Trust Queries). Figure 2 shows an example graph modelling of RM entities without the edges weights, and Figure 5 shows some example interaction graphs with some edge weights.

GI module outputs an interaction graph from the above weighted graph modelling of RM entities, where the topology is defined by set of edges, and quality of interaction between entities is captured by edge weights: higher edge weight indicates stronger interaction and relationship between the two RM entities. For example, high edge weight assigned to a user-resource edge indicates that they had a satisfactory historical interaction.

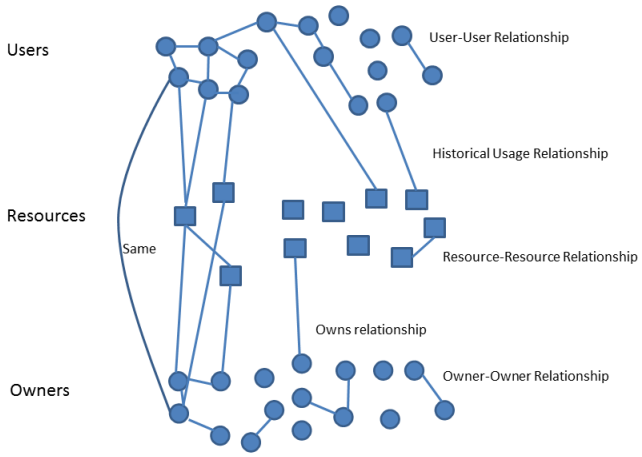



Figure 2: Proposed Interaction Graph Representation.

## 3.2 Spectral Analysis

The Spectral Analysis (SA) module performs trust computation using the interaction graph obtained from GI module. The trust computation is based on an implicit analysis of interaction graph using the spectral graph framework [14]. In contrast to implicit techniques that transform the original graph by projecting it to an alternate multi-dimensional space, explicit techniques use classical graph traversal methods on original graph. There are multiple factors while choosing between explicit and implicit graph analysis technique, e.g., computational complexity, static versus dynamic graph modelling etc. However, implicit techniques, such as the spectral analysis, are more relevant to TRM setup as they enable a topology-centric graph analysis, thereby capturing the overall connectivity between two graph vertices as opposed to shortest path (geodesic metric) or local neighbourhood traversal techniques [15].

The spectral graph framework computes an implicit graph representation known as spectral embedding. The Laplacian embedding [14, 16] is a popular spectral representation technique where each graph vertex is mapped to a K-dimensional space spanned by the first K non-null eigenvectors of the graph Laplacian matrix. The trust index computation, uses the Laplacian embedding of the interaction graph, and it exploits the fact that the Euclidean distances in the embedding space reflects the average connectivity between two graph vertices in the original graph. In particular, for two entities in the RM, their average connectivity in the interaction graph, is used as a measure of level of trust between two entities. Thus, small Euclidean distance between two RM entities in the embedding space reflects a strong average connectivity between them over the interaction graph which is interpreted as high trust level between them.

**Trust Queries handled by SA:** The RM can request two kinds the trust queries to the TRM which are handled by the SA module in TRM.

1. For a given reference user, the RM can ask for trust index (or trust ranking) of various resources or resource owner. The result of this query can be used by the RM to schedule a task from reference user on resources that are well-trusted (i.e., trust index or ranking is above a certain threshold) by the user.
2. For a given reference resource owner, the RM can ask for trust index (or trust ranking) of various users. This user ranking can be used by the reference resource owner to determine which users' tasks are executed on her resources. In particular, for well-trusted users, the resource owner can provide privileged access over its resources (e.g., access to operating system kernel calls).

The first type of resources ranking query is executed by computing the sorted Euclidean distances between projection of reference user and all the resource vertices in the embedding space. In an off-line configuration, TRM precomputes interaction graph representation and stores the pairwise Euclidean distances between each pair of vertices projected in the embedding space as well as sort indices with respect to every user vertex. This step can be periodically repeated in order to consider recent changes in profiles and historical usage database. However, an on-line configuration is also possible where the historical usage information and other profile information is continuously modified by TRM system via DM module and subsequently, for every ranking query a new graph is induced depending on system provided preferences.

A method for computation of trust ranking of resources with respect to a user is as follows:

**INPUT:** interaction graph, embedding dimension(K), set of reference user vertices.

**OUTPUT:** Trust ranking indices of resources.

I Compute a K-dimensional Laplacian embedding of interaction graph.

II For each reference user vertex, compute the Euclidean distance to all the resource vertices in the embedding space.

III For each reference user vertex, sort these Euclidean distances in the increasing order so that smaller distances corresponds to higher ranking.

IV Return ranked indices of resources w.r.t. set of reference user vertices.

As consequence of interaction graph being undirected, the relationships in the graph are bi-directional. Hence, the computation for second type of user ranking query is similar to the first type of query, and it does not require additional efforts except the sorting of the user vertices based on their distances (in the embedding space) from a reference resource vertex.

A method for finding user rankings with respect to a resource owner is as follows:

**INPUT:** interaction graph, embedding dimension(K), set of reference owner vertices.

**OUTPUT:** Trust ranking indices of users.

I Compute a K-dimensional Laplacian embedding of interaction graph.

II For each reference owner vertex, compute the Euclidean distance to all the user vertices in the embedding space.

III For each reference owner vertex, sort these Euclidean distances in the increasing order so that smaller distances corresponds to higher ranking.

IV Return ranked indices of users w.r.t. set of reference owner vertices.

## 3.3 Scalability

The scalability of proposed TRM system has two major aspects. The first aspect deals with scalability of TCE. The traditional bottleneck of spectral (Laplacian) analysis of large graphs is the Eigen-decomposition step which can be handled easily due to the inherent sparseness of Laplacian embedding method which can be efficiently processed with existing sparse eigen-solvers [19] with computational complexity of $O(Cn^2)$ for graph with $n$ vertices and constant $C$ which is directly propotional to number of eigenvectors computed, i.e., $K$.

The second aspect of scalability deals with number of queries posted to TCE module. In this aspect, the proposed system is highly scalable given that, once the spectral embedding is computed, the ranking queries can be efficiently solved by computing and sorting the Euclidean distances in embedding space.

# 4. EXPERIMENTAL EVALUATION

In this section, we provide details of the experimental setup used for interaction graph modelling and associated spectral analysis that enables both qualitative and quantitative evaluation of trust queries.

## 4.1 Interaction Graph Modelling Setup

The interaction graph modelling involves a combination of multiple data sources stitched together namely, large real resource usage traces from grid computing, employee and resource details from internal organizational database of a large IT company and synthesized social media connections.

**Grid Computing Traces (GCT):** Resource usage traces are used from publicly available dataset *The Grid Workloads Archive*, (http://gwa.ewi.tudelft.nl/datasets/gwa-t-1-das2). The traces were generated by DAS2 sytem in 2005 where the system had 5 sites with total of 1124772 jobs submitted by 333 users. We preprocessed this data and kept 1048539 jobs for which a completion status (successful/failed) was known.

**Users & Owners Database (UOD):** We have used an internal organizational database from a large IT company which comprises employees and allocated resource for synthesizing the realistic user and owner data in the setup. This database also contains the employees' location and department which act as useful attributes of users and owners in our setup.

**Resource Configuration Data (RCD):** This is also an internal organizational dataset which details the configuration as well as location of each resource allocated to employees. We used this information to characterize the computational capability as well as location constraint of resources.

**Social Media Connections (SMC):** We generated synthesized social network graph using the (random walk) graph model [18](http://current.cs.ucsb.edu/socialmodels/) where we used following parameter to control the graph structure: 1) $q_e$ : the probability of continuing the walk after each step and 2) $q_v$ : the probability of attaching to visited node.

### 4.1.1 Graph Induction

Data from different sources listed above was combined in following fashion. First, jobs in GCT were assigned to the 214 resources (identified as R-1 to R-433) in RCD randomly and depending on the success and failure of the job the feedback was calculated. If job was successful feedback values were 3 to 5 otherwise it ranged from 1-3. Total number of jobs used were 1048539, out of which only 1125 were failed jobs. We had total 433 users (identified as U-1 to U-433) from UOD, among which 214 were also owners (identified as R-1 to R-214) of the resources where each owner had exactly one resource allocated from RCD. The final social media graph was generated for all user and owners yielding total of 433 nodes and 5083 edges with parameter values $q_e = 0.9995$ and $q_v = 0.0035$. Figure 3 shows the social media graph induced on all users/owners nodes.

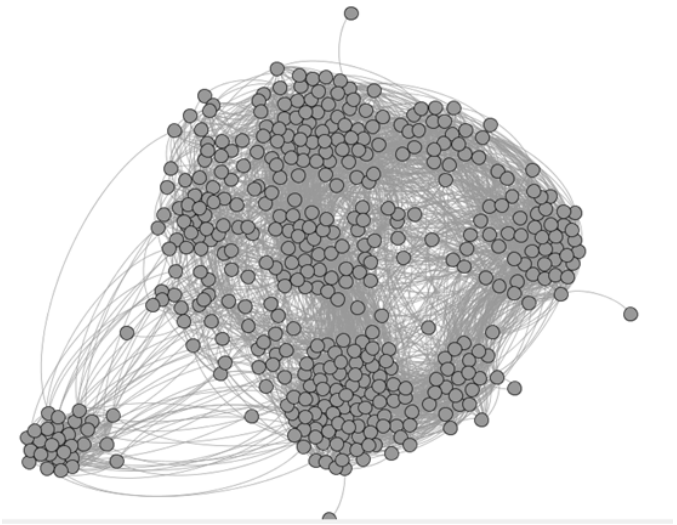


Figure 3: A visualization of social meda conenction graph induced on all users and owners.

Table 2 gives weighting parameter used in the edge weights functions listed in Table 1. Finally, the overall combined realistic traces were fed to graph induction algorithm outlined in Section 3.1 to induce an interaction graph. The output interaction graph had total 861 nodes comprising 214 resources nodes, 214 owners (who were also users) and 219 users. Total number of edges with non-zero edge weights were 70317 where each edge weight is normalized between 0 & 1 as enforced by associated functions.

| Func. | Parameter 1 | Parameter 2 | Parameter 3 |
|---|---|---|---|
| $f_{uu}$ | $social \sim 0.2$ | $location \sim 0.4$ | $org. \sim 0.4$ |
| $f_{oo}$ | $social \sim 0.2$ | $location \sim 0.4$ | $org. \sim 0.4$ |
| $f_{rr}$ | $location \sim 0.3$ | $capability \sim 0.1$ | |
| $f_{or}$ | $constant \sim 0.3$ | | |
| $f_{uo}$ | $constant \sim 0.3$ | | |

Table 2: Weights used to combine parameters of edge weight functions.

## 4.2 Spectral Analysis Setup

We consistently used $K = 10$ as the only parameter for spectral analysis module. This is a significantly low value for the Laplacian embedding dimension as $K \ll n$. This effectively reduces the overall computational complexity of the method as discussed in Section 3.3. We have developed a java based application interface which enables an admin-

istrator to induce an interaction graph and perform spectral analysis.

### 4.3 Evaluation of Trust Queries

Evaluation of trust is not straightforward given the absence of notion of ground-truth. Additionally, the output trust index is subsequently fed to the Resource Marketplace scheduler where while achieving a trade-off between trust and SLA requirements, it is possible sometime that SLA requirement might take priority over trust index. In this work, first we perform qualitative evaluation where we visualize the respective graph structure with reference node and top ranked nodes. Later, we perform a quantitative evaluation of robustness aspect of trust queries.

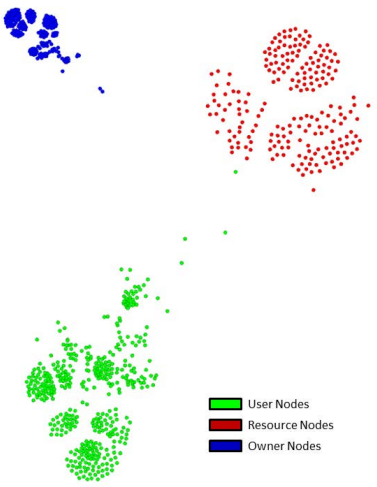
Figure 4 shows different aspects of qualitative trust index evaluations plotted in open-source graph visualization tool Gephi [17]. First, the induced interaction graph without edges is visualized in Figure 4a where different colors are used to characterize different types of nodes (see the figure legends). Next, the same interaction graph is visualized with edges where annotations are attached to different sets of nodes in Figure 4b. In particular, a user node U-85 is annotated as the reference node for which we computed trust index of all resources. The five top ranked resource nodes are also annotated in the same figure with R-23 being the resource with highest trust ranking. This resource belong to owner O-23. Later, we computed the owner to user ranking by taking O-23 as the reference owner node and associated top ranked users are shown in Figure 4b where user U-85 being the top ranked user. This case is one of the case where the bi-directional trust rankings are symmetric thereby giving a higher confidence to the scheduler that user and resource owner both trust each other.

The proposed trust computation is also robust in the sense that it considers average connectivity over graph topology as opposed to individual paths in the (local) neighbourhood. Thus, a trust ranking is robustly derived based on overall interaction between users, resources as well as resource owners in RM. This topology-centric analysis that considers overall interaction handles (to a certain extent) scenarios where some users/owners try to cheat or mislead the trust computation system by adding social media connections to artificially inflate weights of few edges. Figure 5 summarize the robustness analysis of bi-directional trust indices. First we computed top $\alpha$ (= 5,10,15 and 20) ranked resource nodes for each user node for the original graph as well as graphs modified by adding random pairwise edges among owner nodes (varying between 1-20). Later, we computed the retention by finding the intersection of original and modified rankings and plotted the average % retention for each $\alpha$ as shown in Figure 5a. Similarly, we plotted the average % retention for all $\alpha$ in case of owner-to-user trust rankings in Figure 5b. We can observe that retention values are quite high which suggest that artificially inflating the edge weights for minority edges in the interaction graph can't significantly change the trust rankings.

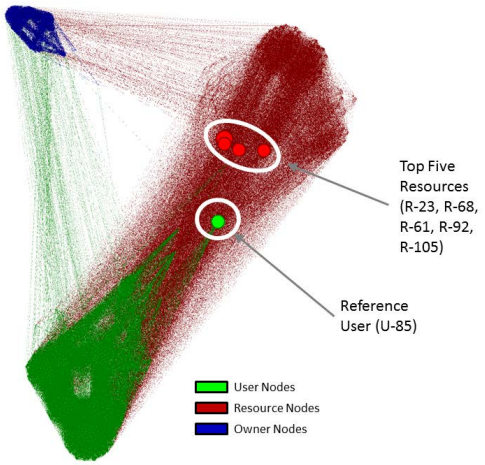
Thus, to significantly influence the trust indices computed using the methods proposed in this paper, the cheating entities need to influence many edge weights, which may be challenging for a single or a small group of entities.

## 5. CONCLUSION

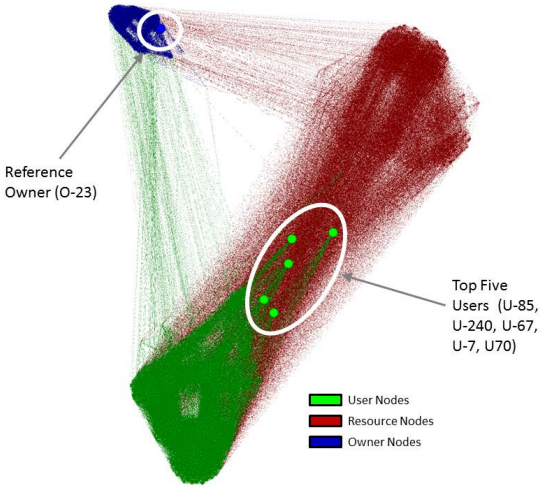
This paper proposes a Trust for Resource Marketplace




(a) Interaction graph visualization without edges.

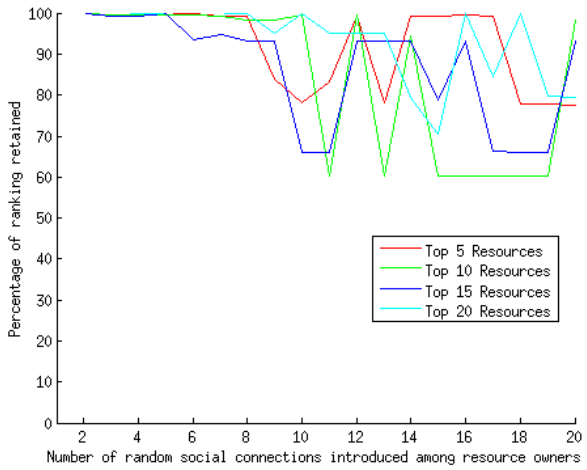



(b) Interaction graph with emphasized reference user node and respective top five ranked resource nodes.

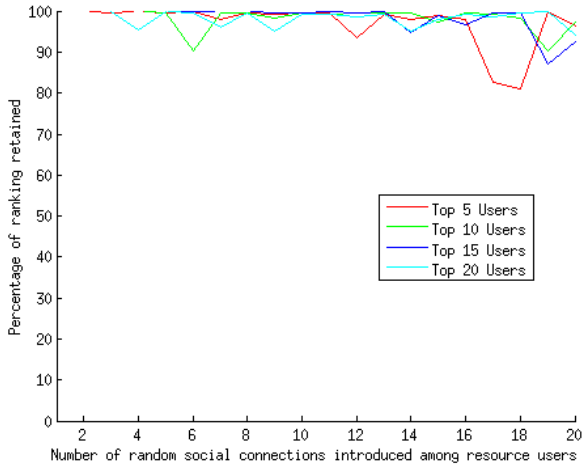



(c) Interaction graph with emphasized reference owner node and respective top five ranked users nodes.

Figure 4: Qualitative Trust Query Evaluation.




(a) Retention plot of user-to-resource trust queries.




(b) Retention plot of owner-to-user trust queries.

Figure 5: Quantitative Evaluation of Trust Queries.

(TRM) system that computes the trust level among the entities in an RM, by incorporating various aspects of the interactions among these entities. In particular, an RM has three kinds of entities: users (with task requests), resources (on which task is executed), and resource owners. Over these entities, the system allows two kinds of trust queries: (i) for a user, a trust indexing of resources or resource owners and (ii) for a resource owner, a trust indexing of users. This is achieved by a novel interaction graph modelling followed by spectral analysis of this graph thereby capturing both direct and indirect relationships among RM entities while deriving trust indexes. Finally, both quantitative and qualitative evaluation of trust queries is performed using interaction obtained from data synthesized from real traces.

## 6. REFERENCES


[1] T. Mukherjee and S. Gujar. *A System to Enable Novel Marketplace for Democratizing Computing Infrastructure*. US Patent Application No. 20130170US01, 2013.

[2] P. Dutta, T. Mukherjee, S. Gujar, and V. Hegde. *C-Cloud: A Cost-Efficient Reliable Cloud of Surplus Computing Resources*. IEEE CLOUD, 2014.

[3] S. M. Habib, V. Varadharajan, and M. Muhlhauser. *A framework for evaluating trust of service providers in cloud marketplaces*. ACM Symposium on Applied Computing (SAC), 1963-1965, 2013.

[4] S. M. Habib, S. Ries, M. Muhlhauser and P. Varikkattu. *Towards a trust management system for cloud computing marketplaces: using caiq as a trust information source*. Security and Communication Networks, (2013).

[5] http://www.cpusage.com/

[6] http://boinc.berkeley.edu/

[7] P. Langhans, C. Wieser, and F. Bry. *Crowdsourcing MapReduce: JSMapReduce*. International conference on World Wide Web companion, 2013.

[8] S. D. Kamvar, M. T. Schlosser and H. Gracia-Molina. *The Eigen Trust Algorithm for Reputation Management in P2P Networks*. 2003.

[9] A. L. Beberg and V. S. Pande. *Storage@home: Petascale distributed storage*. IPDPS 2007.

[10] K. Chard, S. Caton, O. Rana and K. Bubendorfer. *Social Cloud: Cloud Computing in Social Networks*. IEEE Cloud Computing (CLOUD), 2010.

[11] C.M. Tam, P. S. Gill and B. S. Gill. *System and Method for Creating Secure Trusted Social Network*. US Patent US-2006/0259957-A1.

[12] H. Hinton, D. Falola, A. Moran and P. Wardrop. *Method and System for Enabling Trust Infrastructure Support for Federated User Life Cycle Management*. US Patent US-2006/0021018-A1.

[13] T. P. Pannu, E. J. Chen, P. Gilliam and M. Raley. *System and Method for Developing and Using Trusted Policy Based on a social model*.US Patent US 20140245382-A1.

[14] F.R.K. Chung. *Spectral graph theory*. American Mathematical Society, 1997.

[15] H. Qiu and E. R. Hancock. *Clustering and Embedding Using Commute Times*. Pattern Analysis and Machine Intelligence, vol. 29, no. 11, 1873-1890, 2007.

[16] M. Belkin and P. Niyogi. *Laplacian eigenmaps for dimensionality reduction and data representation*. Neural computation, vol. 15, no. 6, 1373-1396, 2003.

[17] M. Bastian, S. Heymann and M. Jacomy. *Gephi - an open source software for exploring and manipulating networks*. International AAAI Conference on Weblogs and Social Media. 2009.

[18] A. Sala, X. Zhao, C. Wilson, H. Zheng, and B. Y. Zhao. *Sharing Graphs using Differentially Private Graph Models*. ACM SIGCOMM (Internet Measurement Conference IMC), 2011.

[19] R. B. Lehoucq , D. C. Sorensen , C. Yang. *ARPACK Users Guide: Solution of Large Scale Eigenvalue Problems by Implicitly Restarted Arnoldi Methods*. 1997.