TRUSTNET: A TRUST AND REPUTATION MANAGEMENT SYSTEM IN DISTRIBUTED ENVIRONMENTS

By

HUANYU ZHAO

Bachelor of Engineering in Information Security
University of Science and Technology of China
Hefei, Anhui, China
2006

Bachelor of Science in Management Science
University of Science and Technology of China
Hefei, Anhui, China
2006

Submitted to the Faculty of the
Graduate College of
Oklahoma State University
in partial fulfillment of
the requirements for
the Degree of
DOCTOR OF PHILOSOPHY
July, 2011

TRUSTNET: A TRUST AND REPUTATION MANAGEMENT SYSTEM IN DISTRIBUTED ENVIRONMENTS

Dissertation Approved:

Dr. Xiaolin Li

Dissertation Advisor

Dr. Subhash Kak

Dr. Johnson Thomas

Dr. Mark Weiser

Dr. Mark E. Payton

Dean of the Graduate College

ACKNOWLEDGMENTS

I would like to gratefully and sincerely express my gratitude to Dr. Xiaolin Li for his guidance, understanding, patience, during my PhD studies at Oklahoma State University. His mentorship was paramount in providing a well rounded experience not only in research skills but also in my long-term career goals and, most importantly, in my life attitude.

I would like to thank Dr. Subhash Kak, Dr. Johnson Thomas and Dr. Weiser, Mark for being my PhD dissertation committees. They have been always there to listen and give me valuable advice. I am deeply grateful to them for the discussions and guidelines that helped me sort out and technical details and the career of my life.

I am very grateful for the friendship of all the members of the Scalable Software Systems Laboratory, especially Han Zhao, Bo Xu, Xinxin Liu, Xin Yang, Rui Yang. These co-workers and friends with whom I worked closely provided for much needed entertainments in what could have otherwise been a somewhat stressful laboratory environment.

I would like to thank the Department of Computer Science at Oklahoma State University. Especially, I would also like to thank Dr. John P. Chandler, Dr. Nohpill Park, Dr. Gopal Rao, Dr. Venkatesh Sarangan, Mr. Terry Wright, Ms. Beau Turner for their assistance in getting my graduate career started and providing me guidance throughout my PhD study.

I express my special thanks to Dr. Sunny Choi and Dr. David Cline for their effort in organizing Oklahoma State University ACM chapter events and activities. We had a great time in running the ACM organization.

Finally, and most importantly, I would like to thank my wife Lianfan Su. Her support, encouragement, unwavering love are in the end what made this dissertation possible. My parents receive my deepest gratitude and love for their dedication and the many years of love and support. It was under their watchful eye that I gained so much drive and an ability to tackle challenges head on.

TABLE OF CONTENTS

LIST OF TABLES

Table Page

LIST OF FIGURES

CHAPTER 1

INTRODUCTION

Because of its salient features of extreme scalability, flexibility, self-configuration, self-organization, and resilience to failures, Peer-to-Peer (P2P) overlay network paradigm has been hailed in both industry and academia [1]. P2P systems have made tremendous progress in fundamental data lookup and content management [2, 3, 4, 5, 6, 7, 8]. The increasing popularity of P2P systems, social networks, online business paradigms (e.g. eBay) has made them prone to malicious behaviors and attacks. Furthermore, many P2P systems do not have central administration and peers are autonomous, making them inherently insecure and untrustful.

In the meanwhile, a large percent of Internet users now routinely use online social networks such as Facebook to manage and document their relationships to others. While such systems provide a convenient and user-friendly interface for users to record aspects of their trust relationships, users are subject to the need to also trust the social network provider, because all messages and data are subject to the provider's policies. Recent user backlashes due to changes in privacy in major providers such as Facebook show that there is significant user demand for social networking infrastructures where users not only establish trust on who they link with, but also who may relay or store data used in the interactions with trusted peers.

To handle trustworthiness issues of these services in open and decentralized environments, trust and reputation scheme has been proposed to establish trust among peers in P2P systems [9]. In a reputation system, statistics are maintained about the performance of each entity in the system, and these statistics are used to infer how

entities are likely to behave in the future. In general, one good P2P reputation system should be low-cost to build, easy to update, and fast in reputation aggregation, inference and dissemination.

However, these schemes focus mainly on how to design fair trust ratings, manage trust locally or globally, and different trust inference mechanisms. There is still no well designed standard and criteria in trust ratings schemes. The current trust ratings schemes are mostly algorithms rather than systems, which are far away from deploying in real P2P and social networks. Different from these relatively isolated studies, we aim to propose a trust and reputation system to promote trust into a coherent trust system with salient features such as end-to-end trust, trust flow, trust spanning tree, and trust closure. Furthermore, this project investigates how to automatically map information inferred from TrustNet into services layered on top of a P2P or social network. To fill the gap between theoretic trust rating schemes and reality P2P application requirements, we propose the notion of trust overlay network middleware (TrustNet) and protocols to initialize a common framework for trust and reputation management research. TrustNet defines the architecture, protocols and services to maintain such an overlay network. Different from most previous work on trust and reputation management systems, we attempt to formulate a set of trust network protocols so that we can have a common framework to investigate, compare, and implement different trust rating and aggregation algorithms, in a similar way as standardized network protocols that paved a solid ground for intensive network research. In TrustNet, we define our own trust rating, trust aggregation, and trust dissemination schemes. TrustNet provides a flexible trust overlay network that can be embedded into existing systems or independently coexist with other distributed systems (e.g. data/content sharing systems and social network). Based on TrustNet, applications or upper-level middleware can obtain all trust services and also provide feedback on transactions.

Our approach building upon Peer-to-Peer computing (P2P) and ad hoc networks applies not only to data lookup and content/information retrieval, but also to autonomously map social network relationships. We aim to provide a communication layer upon which existing as well as emerging social "friend-to-friend" applications that address privacy and trust concerns in a decentralized manner. We pursues research that lay the foundation of a transformative architecture of peer-to-peer overlays that are able to self-organizing trust models which can be effectively mapped to access control in end-to-end communications and services.

Research Goals. *Our overarching research goal in this proposed project is to formulate a TrustNet framework with well-defined protocols and algorithms towards unified trust network research and standardization.* TrustNet is aimed to initiate pioneering trust network research and theory, filling the gap of the missing science and common foundation in the current trust and reputation community. The success of TrustNet initiative could lead to a paradigm shift in trust and reputation management research. To meet these trailblazing goals, we propose the following research thrusts.

- Define the TrustNet overlay network architecture and initiate its specification blueprint. What is the best architecture for trust dissemination, aggregation, inference, and management in an efficient, effective, and flexible manner?

- Define trust dissemination and aggregation protocols and algorithms. How to securely and efficiently communicate trust? How to represent trust? How to interpret trust? And how to infer trust? We propose an initial trust vector dissemination protocol.

- Implement and evaluate the proposed TrustNet overlay network architecture and protocols and algorithms on each layer.

- Design the incentive mechanism to reinforce users to provide truthful feedback in reputation systems.

- Build and simulate TrustNet system. The prototype can start with a distributed P2P-based computing-resource or social network applications.

The rest of the proposal is structured as follows. Chapter 2 presents the related works for both reputation systems and truthful feedback incentive mechanisms. In Chapter 3, we propose the TrustNet framework and discuss the possible research tasks. We propose our design in Chapter 4 and discuss trust management problems in Chapter 5. Chapter 6 presents the simulation results and Chapter 7 concludes the dissertation.

CHAPTER 2

RELATED WORK

Driven by urgent needs in commercial offerings and increasing popularity for content and multimedia sharing, many trust and reputation systems have been proposed recently, for example, the most popular reputation system is the feedback scheme used by the eBay. Clients give feedbacks after their transactions with a 1, -1 or 0 ratings. eBay is a centralized global reputation system. In eBay, there is a reputation server to store and manage all the entities' reputation. The eBay feedback system assumes most users are providing truthful feedbacks.

Xiong and Liu developed PeerTrust, a reputation-based trust supporting framework in a distributed P2P network [10]. PeerTrust uses five trust parameters to compute trust scores of peers. In their paper, a general trust metric on how to use these parameters to compute global trust ratings is presented.

The EigenTrust scheme proposed by Kamvar et al. proposes a method to compute a global trust value for each peer in the network by calculating the eigenvector of a normalized local trust rating matrix [11]. EigenTrust is inspired by PageRank and it gathers the entire systems history transaction and trust information to yield a global reputation score for each peer. They have designed several threaten models in their paper. The simulation under various attack models shows that EigenTrust significantly decreases the risk.

Zhou et al. developed the PowerTrust system for distributed hash table based P2P structures [12]. The scheme gathers locally peer feedback information and aggregates these scores to the global reputation rating. PowerTrust dynamically selects a small

set of power nodes that are most reputable to increase the reputation aggregation accuracy and computation speed. Zhou et al. have also built the GossipTrust system by gossip-based aggregation algorithms [13]. GossipTrust use bloom filters to achieve efficient storage for the rank of global reputation. These two schemes focus on the reputation aggregation processes. The truthful reporting problem is not the main consideration in their works.

Song et al. built the FuzzyTrust System [14]. They constructed a P2P reputation system based on fuzzy logic inferences. They evaluated the FuzzyTrust system using eBay real transaction data. In the experiment, they compared the performance of FuzzyTrust and EigenTrust and concluded that FuzzyTrust is efficient and robust.

Liang and Shi proposed PET, a personalized economic-based trust model for the P2P resource sharing [15]. PET model consists of two major components: reputation calculation and risk evaluation. Their conclusion is that risk is important in designing a personalized trust system as well as reputation rating.

NICE project is a trust inference scheme in distributed P2P networks [16]. In NICE, the authors model the network as a trust graph. They use the trust inference among the trust graph to infer indirect remote trust. The idea of trust graph and the low overhead trust information search and inference algorithms is the major contribution of this paper. With the direct trust ratings existing, the paper focuses on trust inference instead of the trust rating's truthfulness.

Ali Aydm Selcuk and his colleagues proposed a personalized trust management system to calculate trust rating and identify the malicious peers in a P2P network [17]. In their proposed scheme, the users make record of history transactions. Then the scheme computes trust score by using local history transactions information. The generated trust table will be updated periodically. In the end of their paper, they conducted experimental simulation and concluded that the proposed system was accurate and effective.

H. Zhao and X. Li proposed a H-Index based group trust rating aggregation scheme H-Trust [18] which was inspired by the h-index aggregation technique. In their paper, individual reputation and group reputation is generated by computing the H-value. H. Zhao and X. Li also proposed the concept of trust vector and a trust management scheme VectorTrust for aggregation of distributed trust scores [19, 20], and a trust scheme for cyclic mobile space cTrust [21].

Recently, more reputation systems were proposed. Credence is a decentralized object reputation and ranking management system for large-scale peer-to-peer file shearing networks [22]. The one hop reputation protocol is designed for propagating reputation in P2P network for making service decision [23]. A collaboration-based autonomous reputation system is proposed for Email services [24].

We compared the performance of major trust rating system in [19, 20], and classified some major existing reputation and trust systems into a quadrant of taxonomy in [18] according to rating global [11, 13, 12, 14, 10] or personalized [15, 18, 16, 17, 19, 20, 21] trust rating, full[11, 13, 15, 10] or selective [12, 14, 16, 18, 17, 19, 20, 21] aggregation mechanisms.

In the field of MANETs trust management system, Sonja Buchegger and Jean-Yves Le Boudec proposed a reputation scheme to detect misbehavior in MANETs [25]. Their scheme is based on a modified Bayesian estimation method. Sonja Bucheggery and his colleague also proposed a self-policing reputation mechanism [26]. The scheme is based on nodes' locally observation, and it leverages second-hand trust information to rate and detect misbehaving nodes. The CORE system adopts a reputation mechanism to achieve nodes cooperation in MANETs [27]. The goal of CORE system is to prevent nodes' selfish behavior. [25], [26] and [27] mainly deal with the identification and isolation of misbehaved nodes in MANETs, but the mobility feature of MANETs is not fully addressed in these previous work. Our scheme will focus on high mobility setting (with time and location factors) as well as malicious nodes. Yan Sun

etc. considered trust as a measure of uncertainty, and they presented a formal model to represent, model and evaluate trust in MANETs [28]. Ganeriwal, Saurabh et al. extended the trust scheme application scenario to sensor networks and they built a trust framework for sensor networks [29]. Another Ad Hoc trust scheme is [30] where the trust confidence factor was proposed.

Some works tried to quantitatively model the behavior of honest and dishonest peers in trust systems [31, 32], More recent trust and reputation research addressing various issues include [33, 34, 35, 36, 37, 38]. Comprehensive surveys and overviews can be found in [9, 39, 40, 41].

However, most of the reputation systems assume that users are willing to provide reputation feedbacks voluntarily and truthfully. In the field of truthful feedback incentive mechanisms, to encourage truthful feedbacks, R.Jurca and B.Faltings proposed to compare the two reports from the transacted seller and the buyer [42]. In their scheme, the equilibrium is the cooperative behavior. Their results show that the proposed scheme discovers the real interaction outcome. A side payment approach is used in their incentive scheme.

R.Jurca and B.Faltings also proposed to compare the reports with previous trusted reports to determine the trustworthiness of the report [43]. In this scheme, the equilibria of two incentive-compatible reputation mechanisms is analyzed. By using trusted reports, undesired equilibrium points can be eliminated.

In decentralized environment, R.Jurca et al. proposed a currency-based payment scheme that encourages peers to provide truthful ratings[44]. In their paper, the agents only pay for the reputation feedback if and only if the feedback matches the next feedback submitted for the same target peer from another reporter.

Michal Feldman and his colleague proposed an incentive approach based on game theory [45]. The incentive problem is modeled using prisoners dilemma. A distributed reciprocative decision function is proposed as the incentives technique. However, their

approach did not take lying on the contribution of other peers into account. In our proposed wage-based incentive mechanism, lying will definitely lead to low utility.

A truthful feedbacks incentive scheme for exchanging services in a P2P reputation system was presented by Papaioannou [46]. In their paper, both transacted peers are required to provide feedbacks on the mutual transaction's performance. If the two feedbacks are not consistent which indicates that at least one of them is lying, both transacting nodes are punished. A credibility mechanism is used to determine each peer's punishment severity.

Ze Li and Haiying Shen proposed to use both reputation system and price-based system to ensure cooperative in distributed environment [47]. The authors use game theory to investigate and model the underlying cooperation incentive issues. Their results show that their system that combines both reputation system and price-based system achieves better incentive performance than single reputation system or price-based system.

Most of the above schemes are based on the comparison of the reports [42, 43, 44, 46]. These schemes compare the feedback reports with transacted peers' reports, previous trusted reports, or the next feedback report. This comparison based approach involves the problems in the storage of history reports and the collusion attack risk. Different from the comparison based schemes, our proposed solution does not require peers to verify the information truthfulness. The solution requires only localized wage payment schemes.

E.Fehr and S.Gachter did an interesting experiment in [48]. The results show that individuals are willing to punish selfish behaviors if they are given such chances, although the altruistic punishment is costly for them and yield no material gain. The paper shows that cooperation flourishes in environments where altruistic punishment is enabled.

More related research addressing various incentive issues includes [49, 50, 51, 52,

53, 54, 55, 56, 57].

CHAPTER 3

THE TRUSTNET FRAMEWORK

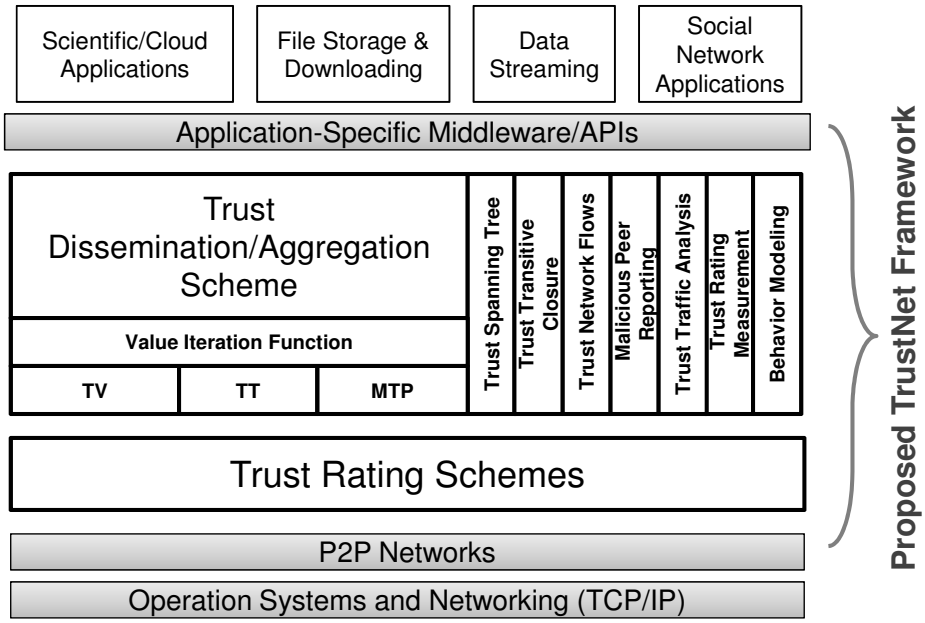


Figure 3.1: TrustNet Conceptual Architecture

As shown in Figure 3.1, the proposed TrustNet architecture is built on top of P2P networks and aimed to provide various trust and reputation services to higher layer applications. The basic foundation of TrustNet is a trust overlay network consisting of *Trust Vector (TV)*. TV is first proposed in TrustNet to represent individual peer's personalized trust towards another peer as a vector of trust direction and trust rating. To maintain such a trust overlay network efficiently and accurately, two sub-layers in TrustNet are proposed. Trust rating layer provides direct trust ratings and trust normalization for direct transactions/file downloads/interation in social networks. Trust dissemination layer define the TrustNet metrics and Trust-

Net dissemination/aggregation protocols. TrustNet provides trusted services to P2P network applications.

Trust dissemination layer defines the TrustNet metrics and most of TrustNet protocols. Trust vector is presented in trust dissemination layer and the relative actions and are also defined. Trust dissemination protocol is running on this layer to provide fast trust propagation for the whole network. Our own trust rating algorithms are designed and implemented in trust rating layer. Especially, we propose the idea of group reputation and group trust. Major existing trust rating schemes should be also implemented in trust rating layer. Trust management strategies and trust models were implemented to manage and evaluate TrustNet in trust dissemination layer. Trust modeling provides a global view to TrustNet manager. Users manage TrustNet and investigate TrustNet features by using TrustNet modeling. All the APIs and interfaces will be defined and provided to TrustNet users. Some initial TrustNet models will be developed with the original version of TrustNet including *Trust Spanning Tree Model*, *Trust Transitive Closure Model*, *Trust Network Flows*, *Malicious Peer Reporting Model*, *Trust Traffic Analysis Model*, *Trust Rating Measurement Model*, *Behavior Modeling Component* and *Incentive Mechanisms*.

TrustNet involves a set of trust services exposed to applications, upper-level middleware, or other coexistent mechanisms. These services include various trust query services, user interfaces to set user priorities, reporting and statistics services. For example, the trust query engine in TrustNet can provide information about a peer's trust rating, a group of peers' trust rating, a peer's outreach range, whether a peer is reachable from another peer, and global statistics.

Trust rating layer gathers the performance of each entity in the direct transactions in a network, and aggregates it into proper trust ratings by using trust rating algorithms. These ratings are used to infer how entities are likely to behave in the future. The trust ratings will spread in the whole network in trust dissemination layer.

The rating scheme should be low-cost to build, easy to update, and fast in reputation aggregation, inference and dissemination.

The foundation of TrustNet research is to lied a solid trust dissemination layer. The trust overlay network is defined in trust dissemination layer. The vertices in TrustNet correspond to peers in a network system. The value of the directed edge A to B reflects how much A trusts B. Each peer maintains a local trust table to store trust information.

The vertices in the graph correspond to peer states (time and location) in a network. An directed solid edge shows the initial trust relation (as trust rating and trust direction) and encounter/contact time. The dashed line shows the peers' movement trace in a social network (In facebook/linkedin where peers have no real movements, the vertices could be considered as peers' various states by time).]

We propose the concept of trust vector in TrustNet. The directed link with trust rating is trust vector. In TrustNet, a trust is propagated as a vector of trust rating and direction, where trust rating is defined as a real number $r, r \in [0, 1]$ and direction is defined as a directed edge in the trust graph. In TrustNet system, we also propose the concept of trust transfer. If Peer A has trust rating $T_{A,B}$ towards Peer B, Peer B has trust rating $T_{B,C}$ towards Peer C, then using trust transfer, A has indirect trust $T_{A,C}$ towards C. $T_{A,C} = T_{A,B} * T_{B,C}$. If A and B have never had a prior transaction, A has to infer a trust value for B by using trust transfer on the trust graph. There might be many trust paths from Peer A to Peer C. Given a set of paths between A and C, A tends to choose the *Most Trustable Path (MTP)*. The most trustable path can be computed as the maximal production value of all directed edges along a path. And this production will be considered as A's trust rating towards C.

For each direct transaction/encounter in the social network system, participating peers generates a direct trust link and assigns a trust rating to represent the quality of this transaction. The link points to the server of this transaction, and the trust rating

could be calculated by different trust rating schemes under various environments. So each transaction in the system can either adds a new directed edge in the trust graph, or relabels the value of an existing edge with its new trust rating or a compound value of both old and new trust ratings calculated by some trust rating schemes.

The trust table is required for each peer. Trust table consist of the remote peer ID as entry, the trust rating for each possible remote peer, the next hop, the total hops (optional)to reach the remote peer and the rating scheme used. Each entry shows only the next hop instead of the whole trust path. Besides of trust table, each peer has to make record for all its previous clients to be used in trust dissemination. The trust table stores only the most reliable trust path.

In the initial stage of an evolving trust overlay network, trust ratings are stored in users' local trust tables. However, the direct or queried trust information is limited and does not cover all potential interactions. For most remote peers, without adequate direct trust information, they have to use indirect trusts to start the process. The inferred trust can be obtained by using *Trust Dissemination Protocol (TDP).*

Our main goal in trust dissemination layer is to develop, formulate, and implement trust dissemination layer protocols. TDP gathers trust ratings to any peer in a network. And spreads and aggregates the trust information. In such protocols, remote and indirect trust information will be added to peers' trust table and be updated as the dissemination process evolves. Two sample dissemination algorithms in TrustNet are VectorTrust Aggregation Algorithm [19, 20] and Markov Chain Based Distributed Trust Aggregation Algorithm [21].

TDP runs on trust dissemination layer to provide fast trust propagation for the whole network by following these four steps: *1) Initialize Trust Table:* Initialize each peer's trust table in TrustNet; *2) Trust Rating:* After each direct transaction or trust query phase, each client peer give trust rating for the server peer and insert this information into client peer's trust table; *3) Periodical Information Update:* Each

peer sends its table to all previous clients neighbor peers periodically; *4) Trust Table Update:* When a peer receives trust tables from its neighbors, it relax the trust rating to all possible peers and updates its own trust table to reflect any changes.

One completed trust dissemination protocol will be proposed, which supports fully dynamic social network system. The protocol will be defined as the rules governing the syntax, semantics, and synchronization of communication. The dissemination protocol should be simple and efficient in terms of the convergence time. The accuracy, scalability, robustness, overhead should be ensured, and the protocol should require little, if any management. It should cover such information as bandwidth, network delay, trust hop count, trust path cost, load, reliability, and communication cost. All the trust dissemination layer protocol detail will be defined, including the handshaking rules. message starting rules, the trust table structure, peer join/leave rules and the message format. The corrupted or improperly formatted messages should be dealt properly (error correction). And detection an unexpected loss of the connection or message will also be considered in our protocol. These schemes can also be further enhanced or replaced by others in TrustNet.

CHAPTER 4

TRUST RATING AND TRUST AGGREGATION

In TrustNet system, we propose three trust dissemination/aggregation scheme to be used in various environments. In highly dynamic environment, we propose the H-Trust aggregation scheme. And in relative static networks, VectorTrust dissemination will be adopted to maintain trust relation. cTrust dissemination leverages the power of trust vector and it provides a solution in cyclic mobile space.

4.1 Reactive Trust Aggregation Scheme: H-Trust

The H-Trust scheme is implemented in five phases. The *trust recording phase* records the information of the past services in service history table which is maintained in the DHT-based overlay network. In *local trust evaluation phase*, a local trust score is calculated by a local trust manager using weighted trust aggregation algorithm. The *trust query phase* is required when the trust information is not available locally. The credibility factors of the responses and H-Index aggregation is proposed in this phase to yield individual's personalized trust rating. The *spatial-temporal update phase* is activated periodically to renew the local trust scores and credibility factors. The most significant and novel phase is the *group reputation evaluation phase* where we propose the first approach to aggregate the group reputation using the H-Trust algorithm. The details of these functional phases are presented in the subsequent sections.

Table 4.1: Local Service History Table

Job ID	**Remote Peer ID**	**Date**	**Service Importance**	**Service Quality**
1001	2	20/07	5	4
1002	4	22/07	2	3
1003	7	24/08	2	5
1004	24	26/08	4	5
...	...	...	...	...

4.1.1 Trust Records

In H-Trust system, we use the Local Service History Table (LSHT) to record the history of past transactions or services information in each peer. Every peer maintains a LSHT for other peers which it had interactions with in the past as illustrated in Table 4.1. The trust rating for one peer towards the other is based on the statistics collected from its LSHT.

The information recorded in the Local Service History Table includes remote peer ID, service date, service importance and service quality. More recent service activities should have a greater impact on a peer's trust score than older ones. The importance of a service is introduced in H-Trust system as an important context that should be incorporated to weigh the feedback for that service. It can act as a defense against some of the subtle malicious attacks where a service provider gains a good reputation by being honest for smaller and less important services and tries to make profit by being dishonest for critical services.

The service quality rating may be a binary value ("0" represents dishonest transaction and "1" represents honest transaction) in some other schemes like Ali's P2Prep [17] or a continuous scale (e.g., [0, 1]). Dellarocas concluded that binary reputation mechanisms will not function well and the resulting market outcome will be unfair if judg-

ment is inferred from knowledge of the sum of positive and negative ratings alone [58]. And the continuous scale is complex for the local peers to rank a service. So we extend the service quality score from binary value and continuous scale to 5 grades rating in a scaled integer of 1 to 5.

Table 4.2: Peer i's Local Trust Rating Table

Remote Peer ID	**Trust Rating**
1	70
2	70
3	40
4	60
...	...
j-1	90
j	N/A
j+1	87
...	...

4.1.2 Local Trust Ratings

Once relevant service history information has been gathered and stored in LSHT, individual peers may use different local inference algorithms to derive trust values. Table 4.2 shows an example Local Trust Rating Table (LTRT) maintained in peer i in the network. Peer i calculates and keeps all the trust ratings towards the other peers of the network locally. We do not limit the Local Trust Rating aggregation functions here. Applying different functions to LSHT allows a peer to calculate a rating best suited for the given situation. Some typical apaches to calculate the local trust ratings are given in [17, 14].

P2P Grid systems involve numerous peers, where a peer often has no direct trans-

action experience with some peers and so that it cannot record all other peers' information in its LSHT. Therefore, it leads to that the trust information towards some peers is not available in LTRT. E.g. the local peer i has no direct transaction experience for remote peer j in peer i's Local Service History Table, so peer i cannot derive trust rating for peer j in its Local Trust Rating Table. This case is illustrated in Table 4.2. In such case, the H-Trust infrastructure has to query other peers for the necessary information over the network, i.e., to activate the trust query phase.

4.1.3 Trust Query

The *trust query phase* must rely on collective opinions from the other peers introduces new challenges. When replying the query, a malicious peer may make false statements about another peer's service due to jealousy or other malicious motives. One trust and reputation system has to accurately filter out such untrustworthy recommendation opinions. Furthermore, the query phase is rather time consuming and possibly incurs heavy overheads when high accuracy or updated reputation is desired. So the major concerns with trust systems are guarantee the validity of opinions and reduce the computing overload.

To address these issues, recommenders' credibility factors must be introduced and critically assessed. The feedback from those peers with higher credibility factors should be weighted more than those with lower credibility. Credibility factors eliminate dishonest recommendation and prevent malicious peers from lying. To reduce the overload, the H-Trust system queries the whole network, yet only consider partially qualified peers' opinion, which are a subset of all the peers.

The credibility of the responses is evaluated according to the past records of the respondents. The results of the past references of the other peers are recorded in every peer's Local Credibility Rating Table (LCRT) as shown in Table 4.3, which is managed as the similar manner as the Local Service History Table. This means there

Table 4.3: Local Credibility Rating Table

Remote Peer ID	**Credibility Rating**
1	7
2	10
3	10
4	5
...	...

are three tables in every local peer: Local Service History Table, Local Trust Rating Table and Local Credibility Rating Table as shown in Figure 4.1. For any new joint peer, the credibility factor is set as 5 (default value).

The query threshold is proposed in H-Trust to reduce the aggregation overload. The query threshold T specifies the boundary of responses to be considered in a queried trust calculation. It is set depending on network environment. When the M responses to a trust query arrive, the querying peer account all the responses. Among them, the N opinions from senders whose credibility rating are greater than T are selected. In our illustrated example, peer i query to the network for peer j's trust rating, and it accounts all the replies. In this example, we set query threshold T=7, and we get the qualified replies as shown by Figure 4.2.

4.1.4 Trust Aggregation

After filtering out opinions from dishonest peers by query threshold T, H-Trust then further aggregates these opinions to yield a proper trust rating. Belonging to the selective aggregation category, it aggregates the selective opinions obtained in the *trust query phase* into local trust ratings.

H-Trust algorithm is inspired by Jorge E. Hirsch's Hirsch Index (H-Index) concept [59]. The h-index is an index that quantifies both the actual scientific produc-

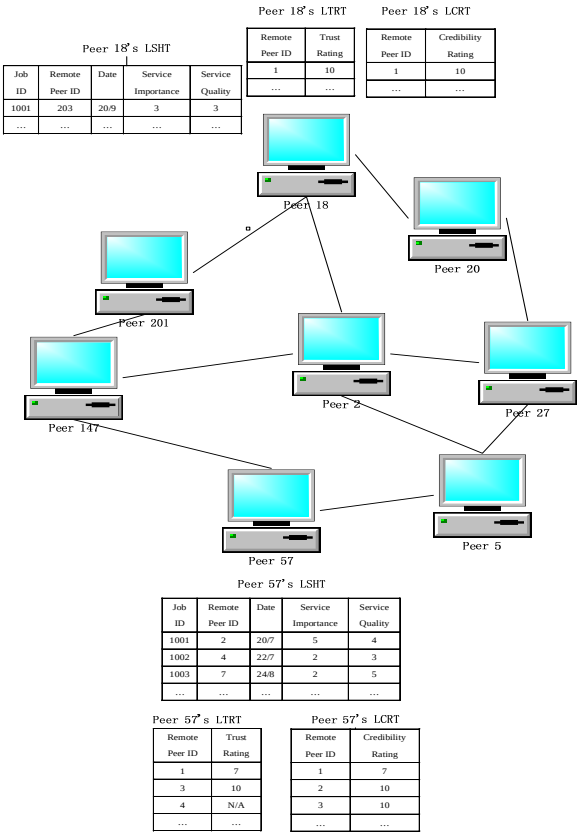


Figure 4.1: Local Peer Record Architecture

tivity and the apparent scientific impact of a scientist, considering the set of the scientist's most cited papers and the number of citations that they have received in other people's publications [60].

We define H-Trust aggregation algorithm as follows. The basic concept is illustrated in Figure 4.3.

Definition 4.1 (H-Trust Aggregation) A peer i has trust rating T_{ij}***=H towards peer j if H of the qualified N peers have at least trust rating score H towards peer j, and the other (N-H) peers have at most trust rating H towards peer j.***

In our illustrated example, peer i first sorts all the obtained qualified replies by trust rating, then it does linear search over the sorted table and finally finds the H-

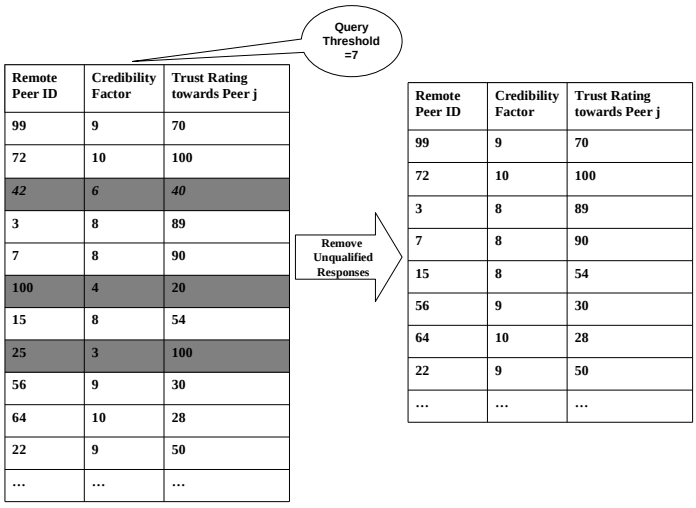


Figure 4.2: Illustrated Example: Quelifitied Responses (Query Threshold=7)

Value equals to 41 as shown by Figure 4.4. One special case needs to be considered. If there is no exactly H-Point, the approximate rank value will always be chosen as H-Value. For instance, if only two peers replied with trust rating 95 and 90. The H-Value should be 2 instead of 90. It is a good way to avoid cheating. It indicates that too few recommendations are not reliable. So only when there are lots of responses and all the replied value is high, H-Value could be high. The detail algorithm to find H-Value is presented in [59].

After the H-Trust aggregation has been done, the H-Value will be set as peer j's trust rating score in peer i's Local Trust Rating Table.

4.1.5 Peer Selection

Once a peer has computed trust ratings for the other peers interested in, it must decide which to choose to start the transaction or service. If there is just only one peer to provide the service, the question is whether to trust it with the service. If multiple peers are offering the same service, the question is whether to select the peer with the highest trust rating. Our solution is that the agent in the H-Trust system may decide based on whether the peer's trust rating is above or below a predefined

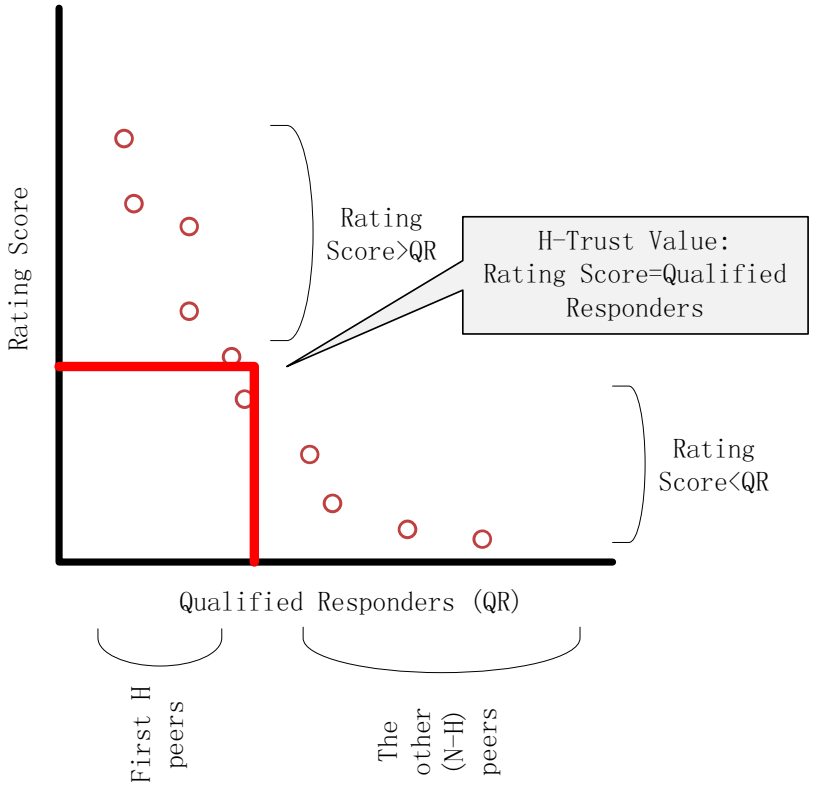


Figure 4.3: H-Trust Calculation

selection threshold T_s which is determined by the service characters.

4.1.6 Update of Trust Rating and Credibility Factors

After each service cycle, the service starter makes record of the service. The record is written back into start peer's LSHT. And The Local Trust Rating Table will be updated. In addition, the starter peer also updates its Local Credibility Table. Note that, in trust query phase, M peers replied the query. No matter whether one peer's response was selected to use or not. We increase the credibility factor of peers who gave correct recommendation, and decrease the credibility of peers who gave incorrect rating in Local Credibility Table.

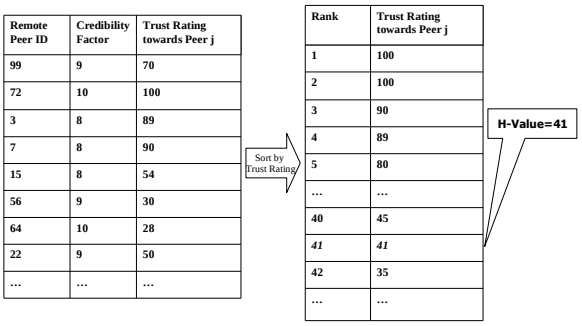


Figure 4.4: Illustrated Example: H-Trust Aggregation

4.1.7 Group Reputation Aggregation and Group Selection

In a current network environment, users perform large-scale and cooperative computational applications, and most computing jobs are accomplished by work groups like P2P Grid Systems [61, 62, 63, 64]. However, nearly all the existing trust schemes and reputation systems did not consider the group reputation issues for collaborative applications and resource sharing. To address these issues, we propose the idea of group reputation. And group trust management systems will be implemented in S3Trust system. S3Trust offers a robust reputation evaluation mechanism for both individual and group trusts with minimal communication and computation overheads. To deal with the increasing network size, hierarchical trust scheme are proposed. Most current trust schemes suffers from collusion behavior. Collusion problem should be carefully studied in our proposed scheme.

The H-Trust personalized trust system developed by our group will be integrated into TrustNet to as an alternative choice to provide accurate and fast trust rating. Another goal in trust rating layer is to ensures good extension feature.

The primary purpose of the group reputation is to help a job distributor to decide which available work group to assign the job. In an open environment, any bunch of peers can form a group and provide services. A service distributor needs to distinguish good service groups from bad ones. So group reputation mechanisms have been proposed and implemented to guide such group selection.

The definition of H-Trust group reputation aggregation algorithm is shown as follows (illustrated in Figure 4.5).

Definition 4.2 (Group Reputation Aggregation) A peer i has trust rating T_{iG}=H_G ***towards group G if*** H_G ***of all the*** N_G ***peers in group G have at least trust rating*** H_G***, and the other*** (N_G-H_G) ***peers have at most trust rating*** H_G ***in peer i.***

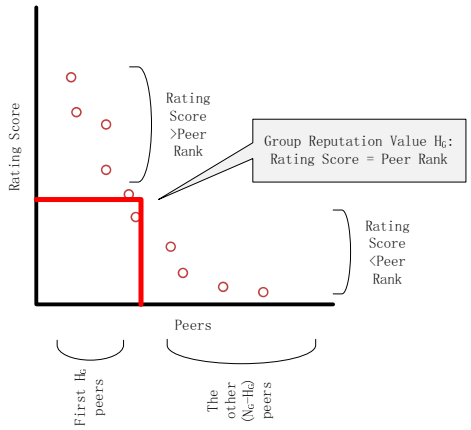


Figure 4.5: Group Reputation Calculation

Peer ID	Trust Rating
6	90
8	100
3	70
7	77
5	54
71	20
22	50
...	...

Sort by Trust Rating

Rank	Trust Rating
1	100
2	90
3	80
4	79
5	77
6	77
...	...
61	*67*
62	62
63	57
...	...

H-Value=62

Figure 4.6: Illustrated Example: Group Reputation Aggregation

In our illustrated example, after the service distributor peer obtain all group members' trust ratings for one group. the service distributor first sorts all the group members by trust rating, then it does linear search over the sorted table and finally finds the H-Value equals to 62 as shown by Figure 4.6. After the group reputation aggregation has been done, the H-Value will be set as the work group's trust rating.

If multiple peer groups offer the same resources, the job distributor would likely go with the peer with the highest trust rating. The group selection can also be made according to a certain criterion such as the quality of service. However, it is important that the selection is not made according to a sole approach, which would result in overloading the trusted peers. Note that, even when many groups are available, an agent may decide to refuse all their service requests if all their reputations lie below the selection threshold T_s. A selection threshold is necessary to protect against malicious spam responses.

4.2 Trust Vector and VectorTrust Dissemination in P2P networks

4.2.1 Trust Vector and Trust Overlay Network

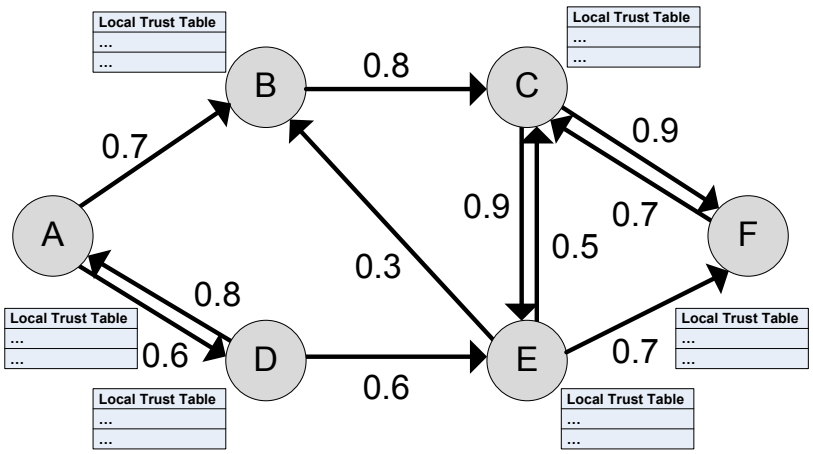


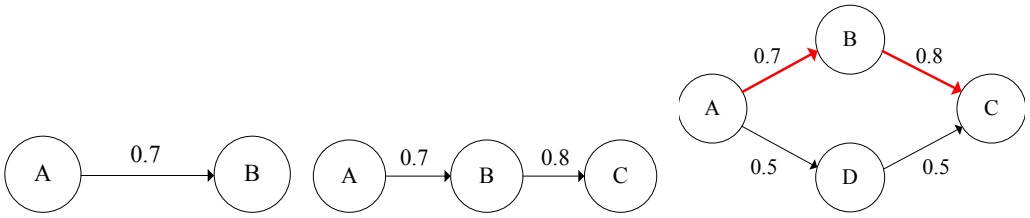
Figure 4.7: Trust Overlay Network. The vertices in the network correspond to peers in a system. An directed edge shows the trust relation as trust rating and trust direction. Each peer maintains a local trust table to store trust information.

VectorTrust system is built on a trust overlay network on the top of a P2P network. Figure 4.7 shows a trust overlay network as a trust graph. The trust graph is a pair $G(V, E)$ where $V = \{v_1, v_2, ...v_n\}$ is the set of vertices (nodes in P2P network), and $E = \{e_1, e_2, ...e_n\}$ is the set of edges (trust relationship). The vertices in the network

correspond to peers in a system. An edge directed from peer A to peer B if and only if peer A was a client for peer B in a direct transaction/interaction and A has a direct trust rating towards B. The value of the directed edge A to B reflects how much A trusts B ($T_{A,B} = 1$ indicates A 100% trusts B, $T_{A,B} = 0$ indicates A never trusts (totally distrust) B).

In VectorTrust, a personalized trust is propagated as a vector of trust rating and direction, where trust rating is defined as a real number $T, T \in [0, 1]$ and direction is defined as a directed edge in the trust graph. This directed link with trust rating is called *Trust Vector* (TV). The formal definition of trust vector is presented as Definition 3.1. If peer A has a trust rating 0.7 on B, the trust vector is $T_{A,B} = 0.7$ as shown in Figure 4.8(a).

Definition 4.3 (Trust Vector): The trust vector is a vector of trust rating and trust direction, where trust rating is defined as a real number $T, T \in [0, 1]$ ***and direction is defined as a directed edge in the trust graph.***



(a) Trust Vector (b) Trust Transfer (c) Most Trustable Path

Figure 4.8: Basic Definition

Note that the edge in a graph represents the rating for a combination of all direct transactions between two peers. The trust rating value could be obtained by applying different functions to consider all the history transactions' importance, date, service quality, etc. How to rate a service and how to generate the accurate direct trust rating are not with the scope of this paper. Some typical approaches to calculate the local trust ratings are given in [18, 17, 14].

Suppose A wishes to find a trust value for C. If A and C had prior transactions,

then A can just look up the value of edge $A \rightarrow C$. However, if A and C have never had a prior transaction, A has to infer a trust value for C by using trust transfer.

Definition 4.4 (Trust Transfer): If peer i has a trust rating $T_{i,j}$ towards peer j, peer j has trust rating $T_{j,k}$ towards peer k, then peer i has indirect trust $T_{i,k} = T_{i,j} \times T_{j,k}$ towards peer k.

As shown by Figure 4.8(b), A has indirect trust $T_{A,C}$ towards C. $T_{A,C} = T_{A,B} \times T_{B,C} = 0.7 \times 0.8 = 0.56$.

There might be many trust paths from peer A to peer C. Given a set of paths between A and C, A tends to choose the *Most Trustable Path* (MTP) to finish multi-hop transactions with an unfamiliar peer C.

Definition 4.5 (Most Trustable Path): The most trustable path from peer i to peer k is the trust path yielding highest trust rating $T_{i,k}$.

In vectortrust, the most trustable path can be computed as the maximal product value of all directed edges along a path. And this product will be considered as A's trust rating towards C. In the example shown in Figure 4.8(c), the MTP is $A \rightarrow B \rightarrow C$, and A infers a trust rating of $T_{A,C} = 0.56$ towards C.

For each direct transaction in the system, participating peers generates a direct trust link and assigns a trust rating to represent the quality of this transaction. For example, consider a successful transaction between users peer A and B in which A is the client of B. After the transaction completes, peer A assigns a trust rating to reflect the quality of B's service. And a new link starts from A with the arrow point to the server B will be added in trust graph. A stores this rating in its trust table. So each transaction in the system can either adds a new directed edge in the trust graph, or relabels the value of an existing edge with its new trust value or a compound value of both old and new trust ratings.

The trust table is required for each peer. As shown in Figure 4.9, it consist of the remote peer ID as entry, the trust rating, the next hop and the total hops (optional)to

Peer D's Local Trust Rating Table

Peer ID	Trust Rating	Via	Path Length
A	0.8	A	1
E	0.6	E	1
B	0.56	A	2
C	0.3	E	2
F	0.42	E	2

Figure 4.9: Sample Trust Table. A local trust table is maintained by each peer. The table consist of the remote peer ID as entry, the trust rating, the next hop and the total trust path length (optional) to reach the remote peer. Each entry shows only the next hop instead of the whole trust path.

reach the remote peer. Each entry shows only the next hop instead of the whole trust path. Besides of trust table, each peer has to make record for all its previous clients ID set ($H'(i)$) to be used in trust aggregation phase.

4.2.2 Trust Vector Aggregation Algorithm

In the initial stage of an evolving trust overlay network, direct trust ratings are stored in local trust tables. However, the direct trust information is limited and does not cover all potential interactions. For most peers without adequate direct trust information, they have to use indirect trusts to start the process. We propose the *Trust Vector Aggregation Algorithm* (TVAA) to infer and aggregate trust values as presented in Algorithm 1. In this algorithm, each trust path is aggregated to MTP with most reliable trust rating towards a target peer by the value iteration

process. Indirect trust information will be added to a trust table and be updated as the aggregation process evolves. Note that, trust aggregation does not create any new link in the trust overlay graph. Links are created or modified only after direct transactions. The value iteration function used in Trust Vector Aggregation Algorithm is given in function (4.9).

$$T_{i,k} = \max(T_{i,k}, T_{i,j} \times T_{j,k}) \tag{4.1}$$

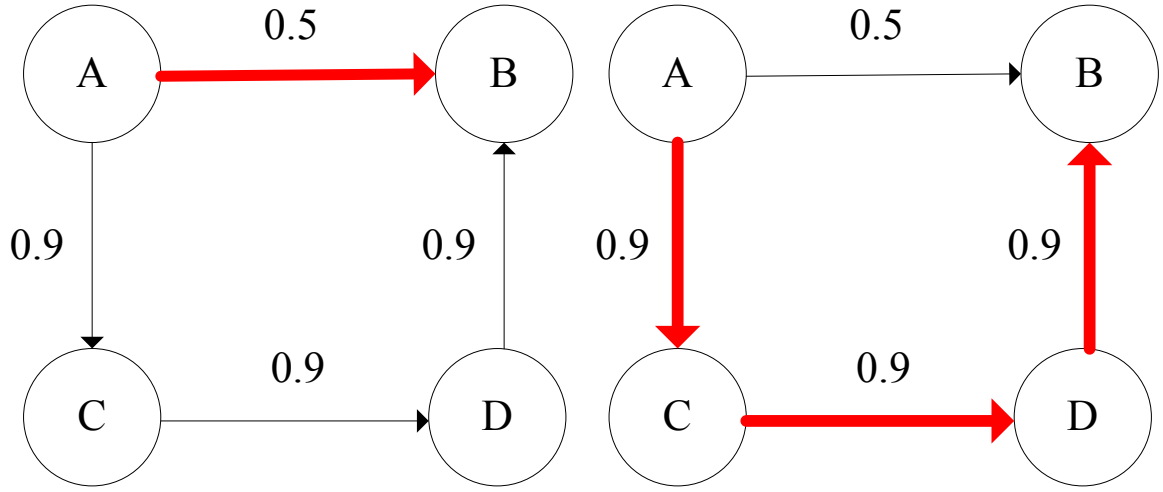
where $T_{i,k}$ is the trust rating towards peer k given peer i's local trust table, $T_{i,j}$ is the direct link trust and $T_{j,k}$ is the received trust information towards peer k.

Algorithm 1 TVAA: Trust Vector Aggregation Algorithm (Peer i's Point of View)

1: Initialize local trust tables.
2: **for** each T_P time **do**
3: Find i's direct precious clients set $H'(i)$.
4: **if** $H'(i) \neq \emptyset$, **then**
5: Send trust table request to those peers.
6: Receive incoming trust tables.
7: Relaxe each trust table entry by trust value iteration function (4.9), update nexthop peers.
8: If receive any trust table request from other peers, send trust table back.
9: **end if**
10: **end for**

The algorithm is implemented based on distributed Bellman-Ford algorithm. Updates are performed periodically where entire or part of a peer's trust table is sent to all its precious clients $H'(i)$ every T_p time. On receiving an update from a neighbor, each peer relax the trust ratings by value iteration function to all possible peers and updates its own trust table to reflect any changes, and then include relevant neighbors

as the next hops.



(a) Direct Trust Path (b) Indirect Trust Path

Figure 4.10: Direct V.S. Indirect Trust Path. In VectorTrust, the direct experience is considered more important because that VectorTrust is a designed as a personalized trust scheme. The peer tends to believe their own experiences more than recommendations.

A special case needs to be considered. If peer A has a direct trust vector to another peer B as shown in Figure 4.10(a), although the trust rating is lower than an indirect path (shown in Figure 4.10(b)). VectorTrust still considers the direct trust as the most trustable path. That is, in VectorTrust, the direct experience cannot be replaced by indirect trust inference. Evidence is not as valuable as direct experience. However, this criterion can be relaxed or modified depending on needs of applications. We consider the direct trust more important because that VectorTrust is a designed as a personalized trust scheme. In personalized trust rating systems, the peer has self-policing trust on other peers, and the peer tends to believe their own experiences more than recommendations.

We present below an illustrated example to ease the understanding of TVAA operations.

Illustrated Example. Consider a network with 6 peers A, B, C, D, E and F

as shown in Figure 4.7. We illustrate the evolution of the trust table in Figure 4.11, where T denotes the current time (or iteration). In the figure, at each iteration, a new trust rating item is highlighted with a darker color.

Peer A' s Local Trust Rating Table

Peer ID	Trust Rating	Via
B	0.7	B
D	0.6	D

Peer B' s Local Trust Rating Table

Peer ID	Trust Rating	Via
C	0.8	C

Peer C' s Local Trust Rating Table

Peer ID	Trust Rating	Via
E	0.9	E
F	0.9	F

Peer D' s Local Trust Rating Table

Peer ID	Trust Rating	Via
A	0.8	A
E	0.6	E

Peer E' s Local Trust Rating Table

Peer ID	Trust Rating	Via
B	0.3	B
C	0.5	C
F	0.7	F

Peer F' s Local Trust Rating Table

Peer ID	Trust Rating	Via
C	0.7	C

(a) Peer A, T=0 (b) Peer B, T=0 (c) Peer C, T=0 (d) Peer D, T=0 (e) Peer E, T=0 (f) Peer F, T=0

Peer A' s Local Trust Rating Table

Peer ID	Trust Rating	Via
B	0.7	B
D	0.6	D
C	0.56	B
E	0.36	D

Peer B' s Local Trust Rating Table

Peer ID	Trust Rating	Via
C	0.8	C
E	0.72	C
F	0.72	C

Peer C' s Local Trust Rating Table

Peer ID	Trust Rating	Via
E	0.9	E
F	0.9	F
B	0.27	E

Peer D' s Local Trust Rating Table

Peer ID	Trust Rating	Via
A	0.8	A
E	0.6	E
B	0.56	A
C	0.3	E
F	0.42	E

Peer E' s Local Trust Rating Table

Peer ID	Trust Rating	Via
B	0.3	B
C	0.5	C
F	0.7	F

Peer F' s Local Trust Rating Table

Peer ID	Trust Rating	Via
C	0.7	C
E	0.63	C

(g) Peer A, T=1 (h) Peer B, T=1 (i) Peer C, T=1 (j) Peer D, T=1 (k) Peer E, T=1 (l) Peer F, T=1

Peer A' s Local Trust Rating Table

Peer ID	Trust Rating	Via
B	0.7	B
D	0.6	D
C	0.56	B
E	0.504	B
F	0.504	B

Peer B' s Local Trust Rating Table

Peer ID	Trust Rating	Via
C	0.8	C
E	0.72	C
F	0.72	C

Peer C' s Local Trust Rating Table

Peer ID	Trust Rating	Via
E	0.9	E
F	0.9	F
B	0.27	E

Peer D' s Local Trust Rating Table

Peer ID	Trust Rating	Via
A	0.8	A
E	0.6	E
B	0.56	A
C	0.448	A
F	0.42	E

Peer E' s Local Trust Rating Table

Peer ID	Trust Rating	Via
B	0.3	B
C	0.5	C
F	0.7	F

Peer F' s Local Trust Rating Table

Peer ID	Trust Rating	Via
C	0.7	C
E	0.63	C
B	0.189	C

(m) Peer A, (n) Peer B, T=2 (o) Peer C, T=2 (p) Peer D, T=2 (q) Peer E, T=2 (r) Peer F, T=2 T=2

Peer A' s Local Trust Rating Table

Peer ID	Trust Rating	Via
B	0.7	B
D	0.6	D
C	0.56	B
E	0.504	B
F	0.504	B

Peer B' s Local Trust Rating Table

Peer ID	Trust Rating	Via
C	0.8	C
E	0.72	C
F	0.72	C

Peer C' s Local Trust Rating Table

Peer ID	Trust Rating	Via
E	0.9	E
F	0.9	F
B	0.27	E

Peer D' s Local Trust Rating Table

Peer ID	Trust Rating	Via
A	0.8	A
E	0.6	E
B	0.56	A
C	0.448	A
F	0.432	A

Peer E' s Local Trust Rating Table

Peer ID	Trust Rating	Via
B	0.3	B
C	0.5	C
F	0.7	F

Peer F' s Local Trust Rating Table

Peer ID	Trust Rating	Via
C	0.7	C
E	0.63	C
B	0.189	C

(s) Peer A, T=4 (t) Peer B, T=4 (u) Peer C, T=4 (v) Peer D, T=4 (w) Peer E, T=4 (x) Peer F, T=4

Figure 4.11: Illustrated VectorTrust Aggregation Example. At time $T = 0$, the original trust tables are constructed according to the direct experiences which are shown by Figure 4.7. The aggregation begins at time $T = 0$. New trust rating items to local trust tables are added with trust propagation process going on. At time $T = 4$, the trust aggregation process stops and results in a convergence status. The convergence speed is 4 time steps/iterations in this case.

At time $T = 0$, the original trust tables are constructed according to the direct experience and are shown in Figure 4.11(a) to 4.11(f). The aggregation begins at

time $T = 0$.

At this point, all peers (A,B,C,D,E,F) send their trust tables to all their previous clients: A to D, B to A and E, C to B, E, and F, D to A, E to C and D, and F to C and E. As each of these neighbors receives this information, they recalculate the trust ratings.

For example, A receives a trust table from B that tells A there is a trust path via B to C, with a trust rating of $T_{B,C} = 0.8$. Since the current trust rating to B is $T_{A,B} = 0.7$, then A knows it has a trust path to C that is $T_{A,C} = T_{A,B} \times T_{B,C} = 0.56$. In the similar way, A learns that it has a trust path to E via D with trust rating $T_{A,E} = 0.36$. As there are no other trust paths that A knows about, it puts these trust ratings into its local trust table as shown in Figure 4.11(g). Similarly, all peers except E have gained new trust information at Time $T = 1$ shown in Figure 4.11(g) to 4.11(l).

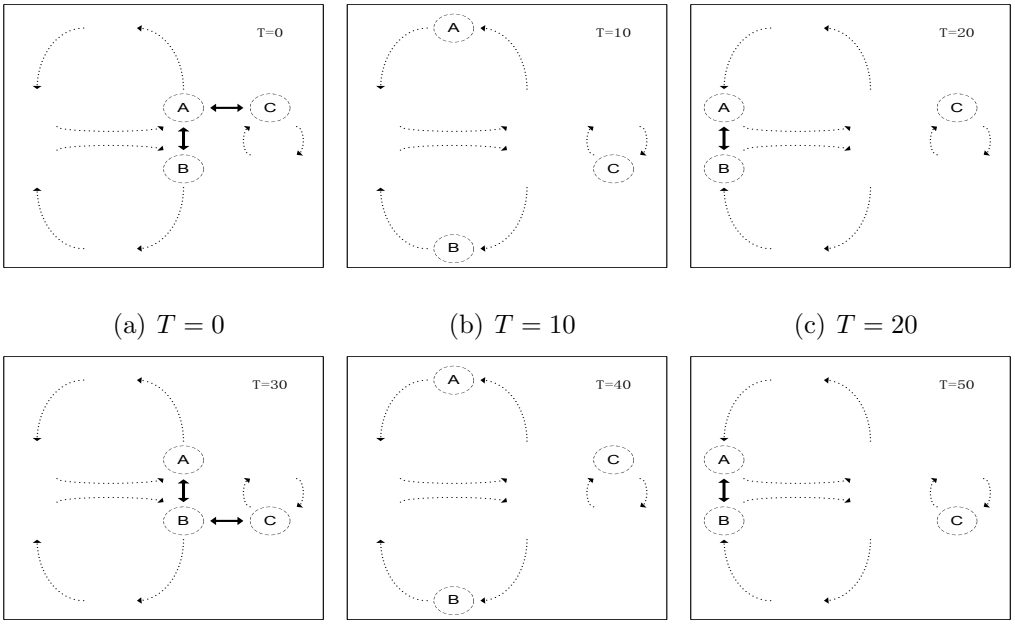
Then all peers broadcast their trust vectors to their previous clients again. This prompts each peer to re-calculate their trust tables. For instance, A receives a trust table from B that tells A there is a path via B to E, with a trust rating $T_{B,E} = 0.72$. Since A's current trust rating to B is $T_{A,B} = 0.7$, then A knows it has a path to E via B that with trust rating $T_{A,E} = T_{A,B} \times T_{B,E} = 0.504$ which greater than the existing trust rating to E via D. At this time, A will replace the old trust rating to E with the new rating $T_{A,E} = 0.504$ via B as shown in Figure 4.11(m).

The process proceeds. At time $T = 4$, none has any new trust information in their tables. Therefore, none receives any new information that might yield any better trust path than it already has. So the algorithm stops and results in a convergence status. Therefore, in this case, the convergence speed is 4 time steps/iterations.

4.3 Trust Aggregation in Cyclic Mobile Environment

4.3.1 Trust Graph in CMANETs

In CMANETs, nodes have short radio range, high mobility, and uncertain connectivity. Two nodes are able to communicate only when they reach each others' transmission range. When two nodes meet at a particular time, they have a contract probability $P(P \in [0, 1])$ that they contact or start some transactions.



(a) $T = 0$ (b) $T = 10$ (c) $T = 20$

(d) $T = 30$ (e) $T = 40$ (f) $T = 50$

Figure 4.12: CMANET Movement Trace Snapshots for One System Cycle ($C_S = 60$)

The cyclic movement trace graph of a CMANET consisting of three nodes is shown in Figure 4.12. The unit time is set as 10. Each peer i moving cyclically has motion cycle time C_i. We can tell from the trace that $C_A = 30$, $C_B = 30$ and $C_C = 20$. The system motion cycle time C_S is the Least Common Multiple (LCM) of all the peers' motion cycle time in the network, $C_S = lcm(C_A, C_B, C_C) = 60$. Note that, CMANETs movement traces are not required to be following some shapes. The "cyclic" is explained that if two nodes meet at time $T0$, they have a high probability to meet after every particular time period T_P. We represent the movement traces in

this paper as some shapes to ease the presentation and understanding.

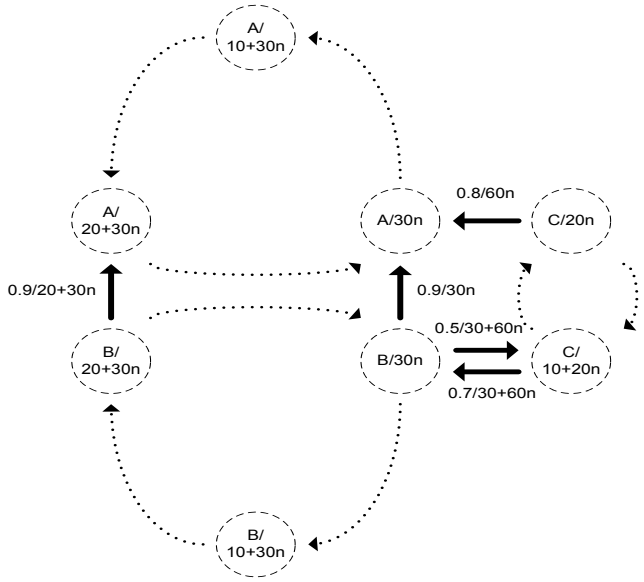


Figure 4.13: Trust Graph. The vertices in the graph correspond to peer states (time and location) in a system. An directed solid edge shows the initial trust relation (as trust rating and trust direction) and contact time. The dashed line shows the nodes' movement trace. Each peer maintains a local trust table to store trust information.

To describe the CMANETs system features and trust relationships, we combine the snapshot graph and trust relationships into a directed trust graph as shown in Figure 4.13. Each node is represented by several states based on periodically appearing locations as the vertices in graph. We represent the states X_i as $i/T_i\{Loc\}$ where i is the node ID and $T_i\{Loc\}$ is appearance time for particular locations. The appearance time is give by,

$$T_i\{Loc\} = T0_i\{Loc\} + C_i \times n (n = 0, 1, 2, \ldots) \tag{4.2}$$

where $T0_i\{Loc\}$ is the first time node i appears at this location and C_i is node i's motion cycle time. For example, node A appears at three locations. So in trust graph, node A is represented by three states: $A/T_A\{Loc_0\}, A/T_A\{Loc_1\}$ and $A/T_A\{Loc_2\}$. Following Equation (4.2), we have, $T_A\{Loc_0\} = 0 + C_A \times n = 30n$, $T_A\{Loc_1\} = 10 + C_A \times n = 10 + 30n$, $T_A\{Loc_2\} = 20 + C_A \times n = 20 + 30n$, $(n = 0, 1, 2, \ldots)$. The

three states generated by node A is $A/30n, A/(10+30n), A/(20+30n)$. State X_i's one hop direct trust neighbors is represented by the set $H'(X_i)$, e.g., $H'(B/30n) = \{A/30n, C/(10+20n)\}$. The directed dashed lines between states shows nodes' movement trace as state transfer edges in trust graph.

The initial trust relationships are shown by the solid directed edges in the graph. There is an edge directed from peer i to peer j if and only if i has a trust rating on j. The value $R_{i,j}(R_{i,j} \in [0,1])$ reflects how much i trusts j where $R_{i,j} = 0$ indicates i never/distrust trust j, $R_{i,j} = 1$ indicates i fully trust j. The trust between different states of the same node i is considered as $R_{i,i} = 1$.

The trust rating is personalized which means the peer has self-policing trust on other peers, rather than obtaining a global reputation value for each peer. We adopt personalized trust rating because in an open and decentralized CMANET environment, peers will not have any centralized infrastructure to maintain a global reputation. The solid trust rating value on the edge could be obtained by applying different functions to consider all the history transactions' importance, date, service quality between two peers. How to rate a service and how to generate and normalize the accurate direct trust ratings are not with the scope of this paper. In this paper, we assume the normalized trust ratings have been generated. What we studied in this paper is the trust aggregation/propagation process in an ad hoc network with high mobility.

Besides trust ratings, each edge is also labeled by a time function showing when two nodes can communicate by this trust link. The appearance time for each link is given by Equation (4.3):

$$T_{R_{i,j}} = T0_{R_{i,j}} + lcm(C_i, C_j) \times n(n = 0, 1, 2, \ldots) \tag{4.3}$$

Where C_i, C_j is the relevant nodes' motion cycle time and $T0_{R_{i,j}}$ is the first time they meet by this link. The solid edges are represented as $R_{i,j}/T_{R_{i,j}}$. The system trust graph shows all the trust relationships, the moving trace and possible contacts

of the network. For example, setting $T = 0$ at Figure 4.13, we obtain the snapshot trace as in Figure 4.12(a) and the appearing trust links.

4.3.2 Trust Path Finding Problems in CMANET

In cTrust system, each peer maintains a local trust table. The trust table consists of the remote peer ID as entry, the trust rating for each possible remote peer, the next hop to reach the remote peer. Each entry shows only the next hop instead of the whole trust path. Initially, peers's trust tables only contain the trust information of their one hop direct experience.

Due to the communication range and power constrains, peers are not able to communicate with remote peers directly. Suppose peer i wishes to start a transaction with remote peer k. i wishes to infer an indirect trust rating for peer k to check k's reputation. In cTrust, the trust transfer is defined as follows.

$R_{i,j}$ and $R_{j,k}$ can be both direct and indirect trust. Beside the trust rating, peer i also wishes to find a trustable path and depend on the multi-hop communication to finish this transaction. Among a set of paths between i and k, i tends to choose the Most Trustable Path (MTP).

MTP is computed as the maximal $\otimes$ production value of all directed edges along a path. And this production will be considered as i's trust rating towards peer k. The MTP provides a trustable communication path, and is used to launch multi-hop transactions with an unfamiliar target peer. cTrust scheme solves the trust rating transfer and MTP finding problems in CMANETs.

4.3.3 Markov Decision Process Model

Markov Decision Process (MDP) is a discrete time stochastic control process consisting of a set of states. In each state there are several actions to choose. For a state x and an action a, the state transition function $P_{x,x'}$ determines the transition

probabilities to the next state. A reward is also earned for each state transition. We model the MTP finding process as a MDP. We propose value iteration to solve the MTP finding problem.

Theorem 4.1: The MTP finding process is a Markov Decision Process.

Proof:

Initially, for a sequence of random node states in trust graph $X_1, X_2, X_3, \ldots, X_t$, the trust path has the following relation:

$$\begin{aligned} &Pr(X_{t+1} = x | X_1 = x_1, X_2 = x_2 ..., X_t = x_t) \\ &= Pr(X_{t+1} = x | X_t = x_t) \end{aligned} \tag{4.4}$$

Equation 4.4 indicates the state transitions of a trust path possess the markov property: the future states depend only on the present state, and are independent of past states.

The components required in a MDP are defined by the following notations:

- *$\mathcal{S}$: state space of MDP, the node state set in the trust graph.*
- *$\mathcal{A}$: action set of MDP, the state transition decisions.*
- *$P_{x,x'} = Pr(X_{t+1} = x' | X_t = x, a_t = a)$: the probability that action a in node state x at time t will lead to node x' at time $t + 1$.*
- *$R_{x,x'}$: the reward received after transition to state x' from state x with transition probability $P_{x,x'}$. $R_{x,x'}$ is in terms of trust rating in out scheme.*

The state transition probability in state x to state x' is computed from normalizing all x's out trust links (trust ratings).

$$P_{x,x'} = Pr(X_{t+1} = x' | X_t = x) = \frac{R_{x,x'}}{\sum\limits_{y \in H'(x)} R_{x,y}} \tag{4.5}$$

In each node state, the next state probability sums to one. The trust path finding process is a stochastic process that all state transitions are probabilistic.

The goal is to maximize the cumulative trust rating for the whole path, typically the expected production from the source peer to the destination peer.

$$\gamma R_{s_1,s_2} \otimes \gamma^2 R_{s_2,s_3} \otimes \gamma^3 R_{s_3,s_4} \otimes ... \otimes \gamma^t R_{s_t,s_{t+1}} \tag{4.6}$$

where γ is the discount rate and satisfies $0 \leq \gamma \leq 1$. It is typically close to 1.

Therefore, the MTP finding process is a MDP $(\mathcal{S}, \mathcal{A}, P_{\cdot,\cdot}, R_{\cdot,\cdot})$.

The solution to this MDP can be expressed as a trust path π (MTP), The standard algorithms to calculate the policy π is the value iteration process.

4.3.4 Value Iteration

Section 4.3.2 presents the trust transfer function $R_{i,k} = R_{i,j} \otimes R_{j,k}$. The upper bound for $R_{i,j} \otimes R_{j,k}$ is $\min(R_{i,j}, R_{j,k})$ because the combination of trust cannot exceed any original trust. $R_{i,j} \otimes R_{j,k}$ should be larger than $R_{i,j} \times R_{j,k}$, which avoid a fast trust rating dropping in trust transfer. The discount rate $\gamma(\gamma \in [0, 1])$ determines the importance of remote trust information. The trust transfer function $R_{i,j} \otimes R_{j,k}$ needs to meet the following condition:

$$R_{i,j} \times \gamma R_{j,k} \leq R_{i,j} \otimes R_{j,k} \leq \min(R_{i,j}, \gamma R_{j,k}) \tag{4.7}$$

In cTrust scheme, we set the trust transfer function as:

$$R_{i,j} \otimes R_{j,k} = \min(R_{i,j}, \gamma R_{j,k}) \times \sqrt[n_a]{\max(R_{i,j}, \gamma R_{j,k})} \tag{4.8}$$

We prove that the given function meets the condition in (4.7).

Proof:

$$\max(R_{i,j}, \gamma R_{j,k}) \leq 1, so, \min(R_{i,j}, \gamma R_{j,k}) \times \sqrt[n_a]{\max(R_{i,j}, \gamma R_{j,k})} \leq \min(R_{i,j}, \gamma R_{j,k})$$

$$\min(R_{i,j}, \gamma R_{j,k}) \times \sqrt[n_a]{\max(R_{i,j}, \gamma R_{j,k})}$$

$$= \frac{\min(R_{i,j}, \gamma R_{j,k}) \times \max(R_{i,j}, \gamma R_{j,k})}{\sqrt[n_a - 1]{\max(R_{i,j}, \gamma R_{j,k})}} = \frac{R_{i,j} \times \gamma R_{j,k}}{\sqrt[n_a - 1]{\max(R_{i,j}, \gamma R_{j,k})}} \geq R_{i,j} \times \gamma R_{j,k}$$

Therefore, we have proved that the trust transfer function (4.8) meets the condition (4.7).

By setting up the adjusting factor $n_a(n_a{=}1, 2, 3...)$, $(R_{i,j} \otimes R_{j,k})$ can be sliding between the upper and lower bound.

In each round of the iteration, the trust table of each node is updated by choosing an action (next hop state in trust graph). The value iteration is executed concurrently

for all nodes. It compares the new information with the old trust information and makes a correction to the trust tables based on the new information. The trust tables associated with the nodes are updated iteratively and until they converge. Based on the trust transfer function, the value iteration function is set up as:

$$R_{i,k} = \max\left(R_{i,k}, \alpha\left[\min(R_{i,j}, \gamma R_{j,k}) \sqrt[n_a]{\max(R_{i,j}, \gamma R_{j,k})}\right]\right) \tag{4.9}$$

where $R_{i,k}$ is the trust rating towards peer k given peer i's local trust table, $R_{i,j}$ is the direct link trust and $R_{j,k}$ is the received trust information towards peer k. $\alpha(\alpha \in [0,1])$ is the learning rate. The learning rate determines to what extent the newly acquired trust information will replace the old trust rating. A learning rate $\alpha = 0$ indicates that the node does not learn anything, and a learning rate factor $\alpha = 1$ indicates that the node fully trusts and learns the new information. At the convergence status of the value iteration, each peer's trust table will contain the trust rating for MTP.

4.3.5 cTrust Distributed Trust Aggregation Algorithm

In the initial stage of an evolving CMANET, pre-set direct trust ratings are stored in local trust tables. However, the direct trust information is limited and does not cover all potential interactions. The distributed trust aggregation algorithm gathers trust ratings to any peer in a network (Algorithm 2). In this algorithm, each trust path is aggregated to MTP with highest trust rating towards target peer. Indirect trust information will be added to trust tables and be updated as the aggregation process evolves. The algorithm is implemented based on distributed Bellman-Ford algorithm. Updates are performed periodically where peers retrieve one of their direct trust neighbors' trust tables and replace existing trust ratings with more higher ones in local trust tables,then include relevant neighbors as the next hops.

Algorithm 2 Distributed Trust Aggregation Algorithm in CMANETs

1: Initialize local trust tables.
2: **for** each time slot **do**
3: **for** peer $i \in$ CMANET **do**
4: Find i's direct trust neighbor set $H'(i)$.
5: **if** $H'(i) \neq \emptyset$, **then**
6: Normalize transition probability by Equation (4.5).
7: Decide one target node j by transition probability in set $H'(i)$
8: Send trust table request to$j$ (The contact between i and j is based on their contact probability P).
9: Receive incoming trust tables.
10: Relaxe each trust table entry by trust value iteration function (4.9), update nexthop peers.
11: If receive any trust table request from other peers, send trust table back.
12: **end if**
13: **end for**
14: **end for**

CHAPTER 5

DISCUSSIONS

5.1 Trust Spanning Tree

In TrustNet, the trust reachability is defined as follows.

Definition 5.1 (Trust Reachability)*: Peer j is trust reachable from peer i if there is a trust path that has higher trust rating than a trust threshold from i to j.*

In a trust graph, directed trust vectors merely show a binary neighbor relationship, for instance, whether Node A is connected to Node B. To have a better understanding on the trust relationships beyond one hop away and gain more insight on the trust landscape, it's desirable for a peer/node to be able to obtain knowledge about its multi-hop outreach scope with some trust constraints. To this end, we introduce the concept of trust spanning tree to represent the outreach scope of a peer to its trustable peers with a pre-defined trust threshold θ.

Definition 5.2 (Trust Spanning Tree)*: In TrustNet, peer i's trust spanning tree is a tree with a maximal set of trustable peers that i can reach. The trust threshold θ limits the minimum inferred trust rating from the root to other peers within the tree.*

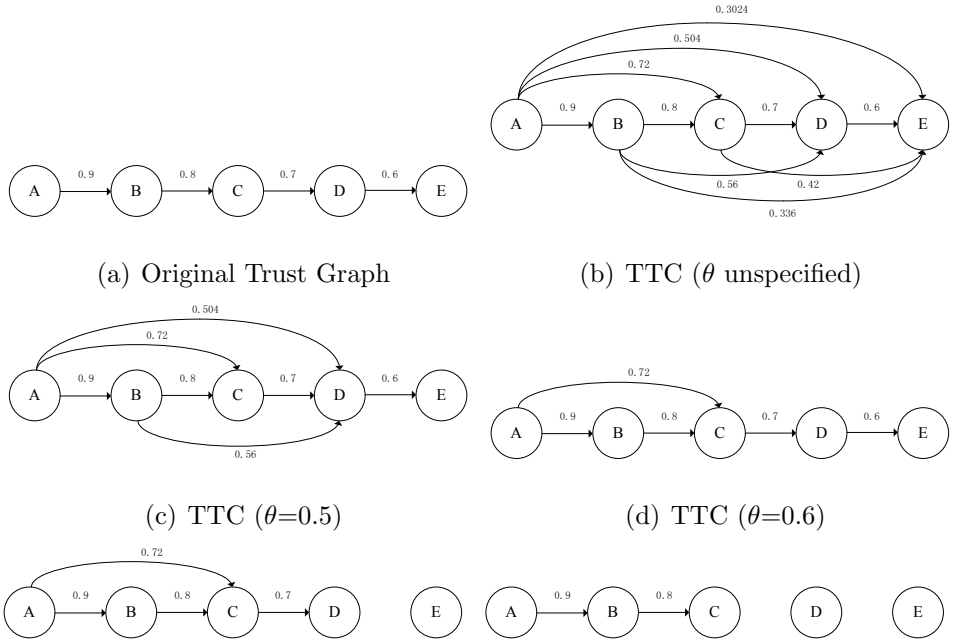
Trust spanning tree can be easily constructed using Depth-First Search (DFS) or Breadth-First Search (BFS).

5.2 Trust Transitive Closure

The trust spanning tree answers the reachability scope problem for each peer in TrustNet. From the system's point of view, the system manager may also wish to

know the reachability characteristic for the entire network. In other words, we want to find out whether there exists a reliable trust path between two randomly picked peers. This knowledge is very useful in designing and improving a trust system. Inspired by the concept of transitive closure in graph theory, we propose the concept of *Trust Transitive Closure*.

Definition 5.3 (Trust Transitive Closure)*: Consider a trust graph G(V,E), where V is the set of peers and E is the set of direct trust links. The Trust Transitive Closure (TTC) with a trust threshold θ of trust graph G is a sub-graph G'(θ)=(V,E') such that for all v,w in V there is a trust link (v,w) in E' if and only if there exists a path from v to w in G and the trust rating $T_{vw} \geq \theta$ in G.*



(a) Original Trust Graph

(b) TTC (θ unspecified)

(c) TTC (θ=0.5)

(d) TTC (θ=0.6)

(e) TTC (θ=0.7)

(f) TTC (θ=0.8)

Figure 5.1: Trust Transitive Closure

We illustrate the TTC concept through an example shown in Figure 5.1. With various trust thresholds, we have different sets of closures. Naturally, we observe that with higher trust threshold, the size of the closure decreases, which reflects the fact that each peer has a relatively smaller highly trustable cliques. The trust transitive

closure reflects the trust connectivity level of a trust overlay network. For a pre-set trust threshold, a network with larger TTC and more trust closure links is generally considered more trust stable than network with small trust closure and trust links. Keeping a network trust connected, the maximum trust threshold is defined as the network's trust threshold. For example, in Figure 5.1, the network's trust threshold is θ=0.6. If we increase the trust threshold to θ=0.7, the network will loss the trust connectivity to peer E, which indicates that peer E is not trustable for trust threshold θ=0.7.

Algorithm 3 Trust Transitive Closure Computation Algorithm

```
1: Represent a trust graph in an adjacency matrix adjmat[max][max]
2: Copy the adjacency matrix into another matrix trust[max][max]
3: for i = 0; i <max; i++ do
4:    for j = 0; j < max; j++ do
5:       if trust[i][j] ≥ θ then
6:          for k = 0; k < max; k++ do
7:             if trust[i][j] * trust[j][k] ≥ θ then
8:                trust[i][k] = trust[i][j] * trust[j][k]
9:             end if
10:         end for
11:      end if
12:   end for
13: end for
```

The data structure of TTC is typically stored as a matrix. That is, if $matrix[i][j] = T$, then it is the case that node i can reach node j through one or more hops with a inferred trust rating T. After the construction of trust transitive closures, in $O(1)$ steps one can determine whether node i is reachable from node j with a trust rating higher than the threshold θ. Our algorithm for computing the trust transitive closure

in TrustNet is derived from Floyd-Warshall algorithm as computing the transitive closure of a graph. The detailed procedure is presented in Algorithm 3.

5.3 Incentive Mechanisms

5.3.1 Problem Formulation

System Model. We assume a homogeneous hash-indexed file sharing P2P system without any centralized trust or centralized infrastructure. No peers are pre-trusted. A feedback based reputation system exists in such a environment. We represent the file download/sharing interactions as transactions. After transactions, peers are required by the reputation schemes to submit their feedback messages such as provider identifier, client identifier, feedbacks (rating and evaluation metrics) to other peers based on their node identifier.

Peer Model. We assume most of the peers are strategic in the system, i.e., peers behave rationally to maximize their own welfare. Peers are able to change their behavior independently. Each transacting client is responsible for submitting feedbacks on performance ("successful" or "failed") of their mutual transaction. To simulate some unpredictable behavior in real P2P network, we allow peers to randomly adopt some irrational reporting activities (regardless of the payoff) with a low probability.

Security Issues. Cryptographic mechanisms such as Public Key Infrastructure (PKI) is used to protect the authentication, integrity and non repudiation of feedbacks. Upon registering in the P2P system, peers are required to create their own certificates based on public-private key pairs. The certificates must be signed by the system, i.e., a certain number of peers in the system. The report message of a peer is encrypted with the secret key and uniquely bonding to its identity. Peers that receive feedback message are able to verify the senders and the integrity of messages. Peers only accept valid feedback messages with a valid signature signed by sender's private key and valid time-stamps.

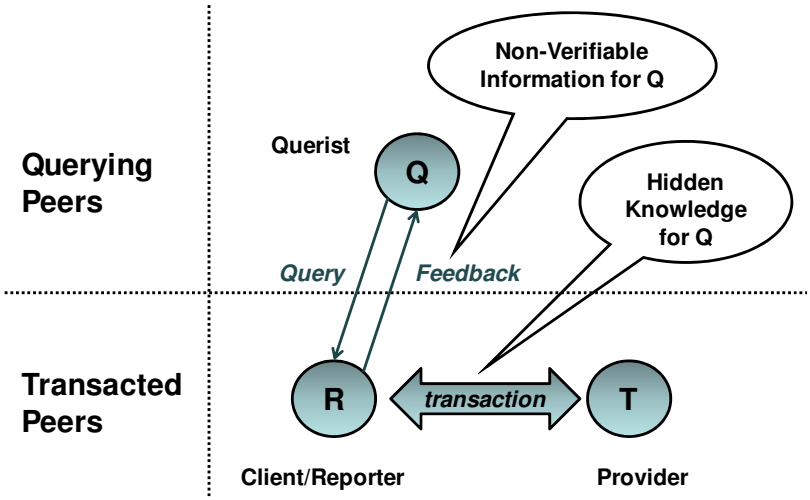


Figure 5.2: The Truthful Feedback Problem for Non-Verifiable Information. For the querist Q, the transaction type is "hidden knowledge". Q issues a query to R to reveal the real transaction type. The feedback from R is non-verifiable information for Q.

Truthful Feedback Reinforcement Problem. As shown in Figure 5.2, the reporter R knows the type ("successful" or "failed") after he finishes the transaction (e.g., file downloading). In the network, a third party peer holding the transaction history records may want to reveal the transaction type to maintain reputation ratings for the transacted peers. For the querist Q, the transaction type is "hidden knowledge". Q issues a query to R to reveal the real transaction type. However, without appropriate incentives, reporter R may not choose to report truth. In addition, the feedback from a reporter R is usually not verifiable. That is, it is hard for querist Q to verify whether reporter R is telling the truth without querying peer T. Even if it queries T, it is still hard for peer Q to judge if R and T are telling the truth. In this case, the objective verification is extremely hard if not impossible. This is the non-verifiable feedback problem/dillema in reputation systems.

The reputation system's basic concern is to obtain the truthful feedbacks on the non-verifiable information environment. To achieve this goal, inspired by the mecha-

nism design paradigm in a hidden knowledge setting [65, 66], we model the feedback reporting process as a reporting game. We design a wage-based incentive mechanism and provide numerical solutions to obtain the minimum wage required to reinforce the truthful strategies. Under our mechanism, querists are not required to estimate/know truthfulness of feedbacks when paying wage. The wage paid to reporters only depends on the feedbacks regardless of truthfulness. By following our scheme, truthful revelation will be a dominant strategy for all reporters. Different from most of the comparison based schemes, our proposed solution does not require peers to verify the information truthfulness. That is, the scheme does not require the peers to compare the feedback submission with other feedbacks. The solution requires only localized wage payment schemes, which greatly reduce the risk of collusion in reporting.

5.3.2 The Game Design

To aid the presentation and analysis, we define the notations to be used throughout the paper as shown in Table 5.1. We assume the output q, the transmission effort c, the querist's utility $\mathcal{U}_Q$, reporter's utility $\mathcal{U}_R$ and social utility $\mathcal{U}_S$ can be evaluated and expressed by a currency (credit) system. The reporter's feedback message is represented by m ($m \in \{successful, failed\}$). P is the probability that a transaction to be successful in a network. It is obtained by surveying the network before we deploy the wage-based incentive mechanism protocol. E.g., in a network where 50% transactions turn out to be successful, P should be set up as $P = 0.5$. We ordinarily assume that reporting nodes have no moral sense. They act to maximize their own welfare. The querying node must design a mechanism that provides incentive for truthful reporting. We model the feedback reporting process as a report game as shown below. In our game, the real transaction type T could be "successful" or "failed". The querist sets up game rules, and it hopes to employ an honest reporting node who always chooses $m = T$. Each agent has the right to choose to play the

Table 5.1: Notations

Notation	**Environment Parameter**
T	The real transaction type, $T \in \{successful, failed\}$
m	Reporter feedback message, $m \in \{successful, failed\}$
w	The wage paid to reporter (w_s for message $m = successful$, w_f for message $m = failed$)
q	Querist output (q_s for message $m = successful$, q_f for message $m = failed$)
α	Cost factor (α_s if $T = successful$, α_f if $T = failed$)
c	Reporter's effort ($c = q/\alpha_s$ if $T = successful$, $c = q/\alpha_f$ if $T = failed$)
P	The probability for one random transaction to be successful
$\mathcal{U}_Q$	Querist utility
$\mathcal{U}_R$	Reporter utility
$\mathcal{U}_S$	Social utility

game or not enter the game. If an agent joins the game as a reporter, it has to obey the game rules. A reporter reports the transaction type as "successful" or "failed". The querist then assigns report wages to complete one reporting cycle. The wage payment function only depends on the report message m.

The Reporting Game

- **Players**

  The querist Q and the reporter R.

- **The Game Procedure**

  - The querist offers the reporter a wage contract $w(q, m)$ (q is output and m is the report message sending by the reporter). The reporter is paid w_s if it reports $m = successful$, w_f if it reports $m = failed$.
  - The reporter accepts or rejects the querist's offer.
  - The transaction state $T = successful$ with probability P and $T = failed$ with probability $(1-P)$. The reporter observes the type of the transaction, but the querist does not.
  - If the reporter accepted the contract, it costs c^2 ($(c = q/\alpha_s)$ to send report message if $T = successful$, $(c = q/\alpha_f)$ to send report message if $T = failed$). The effort is unobserved by the querist.
  - Querist's output is q_s for message $m = successful$ and q_f for message $m = failed$. And the wage is paid.

- **Utility**

  If the reporter rejects the contract, $\mathcal{U}_R = 0$ and $\mathcal{U}_Q = 0$;

  If the reporter accepts the contract, $\mathcal{U}_R = w - c^2$ and $\mathcal{U}_Q = q - w$;

  $\mathcal{U}_S = \mathcal{U}_Q + \mathcal{U}_R$.

At the time of signing a contract, information is symmetric. The reporter does not know what type the transaction will be and what he is going to report at the point which he must accept or reject the contract. But the information becomes asymmetric later after the game begins. Reporter knows the real transaction type which is "hidden knowledge" and non-verifiable information for querist. The problem for the querist is to design a contract that 1) induces the reporter to make a truthful report, 2) is acceptable for both querist and reporter, 3) maximizes querist's own utility.

5.3.3 Contract Solutions

We present solutions on how to construct an optimal contract for a querist to retrieve truthful information observable by a reporter and unverifiable by a querist in the reporting game. The contract is a set of rules that querist designs and reporter accepts in order to convey information from the reporter to the querist. The contract contains a mapping from each possible report to some action (querist's output and reporter's wage in this game) by the querist. In a specific homogeneous system, to ease the deployment, all the reporters and querists share the same contract. However, multi-contacts solutions can also be considered to achieve better performance. The contract is implemented as a feedback message protocol in a reputation system. The contract offer is a mechanism for getting the reporter to truthfully report the transaction type.

Considering the querist, its utility function equals to its output subtracts the wage paid to reporter in both "successful" and "failed" states,

$$\mathcal{U}_Q = [P(q_s - w_s) + (1 - P)(q_f - w_f)] \tag{5.1}$$

The querist aims to maximize its own utility, so it must solve the maximization problem in its utility function,

$$Maximize(\mathcal{U}_Q) \tag{5.2}$$

In this game, the incentive compatibility constraints includes two aspects, the participation constraint and the self-selection constraint. The contract must satisfy one participation constraint so that the reporter will accept the offer. Self-selection constraints are also required so that the reporter will report the truth. For the reporter, the participation constraint is if it tells the truth, its earnings (utility) must be more than or at least equal to 0. Or the reporters will not have motivation to join the game. That is, the reporter hopes to earn some credits by joining the game.

Therefore, the participation constraint can be represented as,

$$\begin{aligned} &P\mathcal{U}_R(m = s|T = s) + (1 - P)\mathcal{U}_R(m = f|T = f) \\ &= P\left[w_s - \left(\frac{q_s}{\alpha_s}\right)^2\right] + (1 - P)\left[w_f - \left(\frac{q_f}{\alpha_f}\right)^2\right] \geq 0 \end{aligned} \tag{5.3}$$

To maximize it own utility, the querist wants to offer the reporter as little wage as possible. So to solve the problem, the participation constraint in Equation (5.3) is lower-bounded to 0,

$$P\left[w_s - \left(\frac{q_s}{\alpha_s}\right)^2\right] + (1 - P)\left[w_f - \left(\frac{q_f}{\alpha_f}\right)^2\right] = 0 \tag{5.4}$$

In the game, the reporter is paid under one of two options, w_s if it reports $m = successful$ and w_f if it reports $m = failed$. To ensure the reporter telling the truth, reporting the false type for a transaction must result in a low payoff. Since the querist cannot verify the message, the contracts itself must induce reporter self-select truth message to report. In this game, the two self-selection constraints are that if the transaction type is "successful", the report's utility sending $m = successful$ must be more than or at least equal to the utility sending $m = failed$ message,

$$\begin{aligned} &\mathcal{U}_R(m = s|T = s) = w_s - \left(\frac{q_s}{\alpha_s}\right)^2 \geq \\ &\mathcal{U}_R(m = f|T = s) = w_f - \left(\frac{q_f}{\alpha_s}\right)^2 \end{aligned} \tag{5.5}$$

And if the transaction type is "failed", the report's utility sending $m = failed$ must be more than or at least equal to the utility sending $m = successful$ message,

$$\begin{aligned} &\mathcal{U}_R(m = f|T = f) = w_f - \left(\frac{q_f}{\alpha_f}\right)^2 \geq \\ &\mathcal{U}_R(m = s|T = f) = w_s - \left(\frac{q_s}{\alpha_f}\right)^2 \end{aligned} \tag{5.6}$$

In reputation system, we care more for "failed" transaction than "successful" transaction. In a system where we do not care for the report truthfulness for the "successful" transaction, the querist is not willing to increase the wage any more than necessary, so the "successful" state's self-selection constraint in Equation (5.5) will be exactly satisfied,

$$w_s - \left(\frac{q_s}{\alpha_s}\right)^2 = w_f - \left(\frac{q_f}{\alpha_s}\right)^2 \tag{5.7}$$

Solving Equation (5.4) and Equation (5.7) yields,

$$w_s = (1-P)q_f^2\left(\frac{1}{\alpha_f^2} - \frac{1}{\alpha_s^2}\right) + \frac{q_s^2}{\alpha_s^2} \tag{5.8}$$

$$w_f = \frac{Pq_f^2}{\alpha_s^2} + \frac{(1-P)q_f^2}{\alpha_f^2} \tag{5.9}$$

Return to the querist utility maximization problem, substituting w_f and w_s, we rewrite Equation (5.1) as,

$$\begin{aligned}\mathcal{U}_Q = P\left[q_s - (1-P)q_f^2\left(\frac{1}{\alpha_f^2} - \frac{1}{\alpha_s^2}\right) - \frac{q_s^2}{\alpha_s^2}\right] \\ +(1-P)\left[q_f - \frac{Pq_f^2}{\alpha_s^2} - \frac{(1-P)q_f^2}{\alpha_f^2}\right]\end{aligned} \tag{5.10}$$

To get the maximum utility, the first-order conditions are,

$$\frac{\partial \mathcal{U}_Q}{\partial q_s} = P\left(1 - \frac{2q_s}{\alpha_s^2}\right) = 0 \tag{5.11}$$

$$\frac{\partial \mathcal{U}_Q}{\partial q_f} = 1 - \frac{2q_f}{\alpha_f^2} = 0 \tag{5.12}$$

Solving them yields,

$$q_s = \frac{\alpha_s^2}{2} \tag{5.13}$$

$$q_f = \frac{\alpha_f^2}{2} \tag{5.14}$$

Finally, the wage combination that reinforces the truthful reporting is given by,

$$w_s = \frac{(1-P)\alpha_f^2\alpha_s^2 - \alpha_f^4(1-P) + \alpha_s^4}{4\alpha_s^2} \tag{5.15}$$

$$w_f = \frac{P\alpha_f^4 + \alpha_f^2\alpha_s^2(1-P)}{4\alpha_s^2} \tag{5.16}$$

The optimal wage is decided by environment parameters α_s, α_f and transaction state T's probability distribution P. With the optimal wage, we can also obtain querist's maximum utility and reporter's utility,

$$\mathcal{U}_Q = \frac{P\alpha_s^2 - P\alpha_f^2 + \alpha_f^2}{4} \tag{5.17}$$

$$\mathcal{U}_R = 0 \tag{5.18}$$

The reporter does not any earn information rents. Statistically, for the reporters' population or a large enough reporting rounds. the overall expected reporters' utility is 0 because of the single participation constraint is lower-bounded to 0 in Equation (5.4). The querists pay the reporters as little as it can be to achieve own maximum utility. However, for a single reporter at specific time, the utility could be more than or less than zero. Reporters will join this kind of game, trying to gain positive utility. To encourage the participation of more reporters, the querists could slightly increase the wage, which leads to $U_R > 0$.

5.3.4 Integration with Reputation Systems

The proposed wage-based incentive mechanism could be integrated to the some existing reputation systems as an additional message feedback protocol layer. Some typical

incentive compatible reputation systems we are currently working on is [18, 19, 20, 21]. The reporting game is implemented as a message feedback protocol in reputation systems. All the peers who joint the game have to follow the protocol format to send a feedback message. Feedback messages failing to follow the protocol will not be accepted by the system. PKI system is used to protect the authentication, integrity and non repudiation of feedbacks. Upon registering in the wage-based incentive game system, each peer is required to create their own certificates based on public-private key pairs. With a certificate, querying peer that receives feedback message is able to verify the identity of a reporter in the game. The querying peer only accepts valid feedback messages from a node within the wage-based incentive mechanism protocol.

In the wage-based incentive mechanism, after transactions, peers are required by the reputation schemes to provide their feedback messages with provider identifier, client identifier and feedbacks signed by their own identifiers. The feedback message includes transaction performance ("successful" or "failed") and additional evaluation metrics. The message cost (c), the output (q) and wage (w) is determined by the reporting protocol. The cost could be energy consumption correspond to a real system, which could be corresponded to distance. That is, for every message the reporter sent, there is energy cost.

The cost, output, wage and peers' utility should be evaluated and represented by a universal credit/currency scheme. The incentive protocol including the feedback message format, the credit/currency schemes, and additional protocol rules are mandatory installed as the clients' plug-ins when the reporters and querists joined the game.

CHAPTER 6

SIMULATION RESULTS

6.1 Simulate H-Trust in P2P Environment

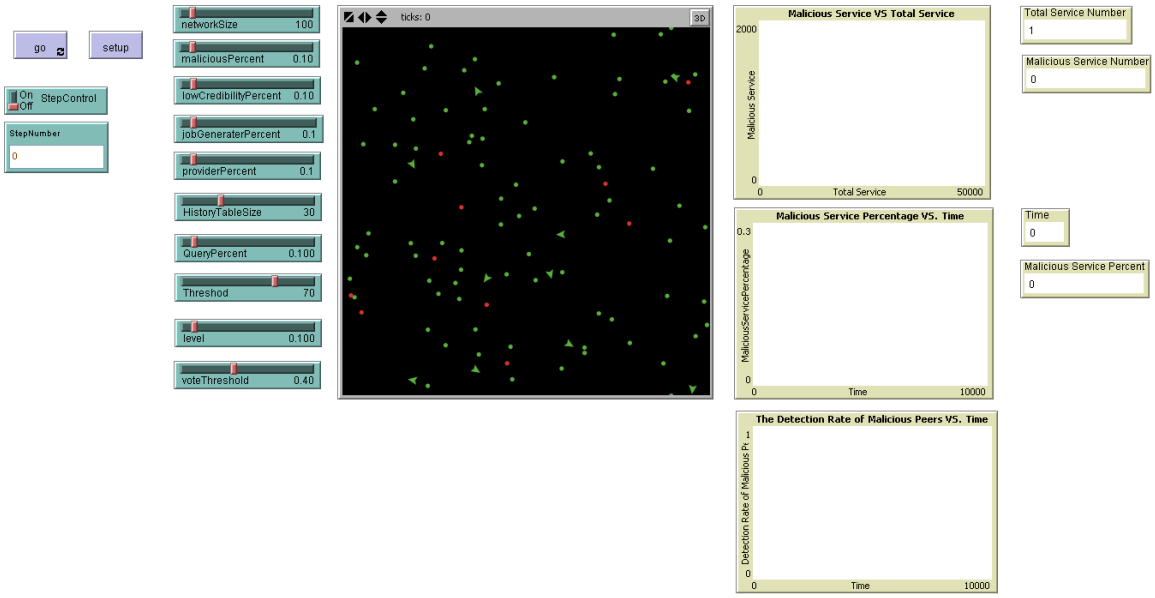


Figure 6.1: User Inferface of the Simulator

In this section, we evaluate the performance of H-Trust through simulation. The simulation is developed using NetLogo, a popular multi-agent simulation tool in the AI community [67]. We use this simulator because its evaluating metrics meet our needs and it was used to model some existing schemes [36]. Thus, using NetLogo allows us to compare our algorithms with others.

The screenshot of the user interface of the simulator is shown in Figure **??**. On the left side, it lists a set of tuning bars that users can configure and adjust system parameters, including network size, percentage of malicious peers, percentage of low-

Table 6.1: Simulation Parameter Setting (H-Trust)

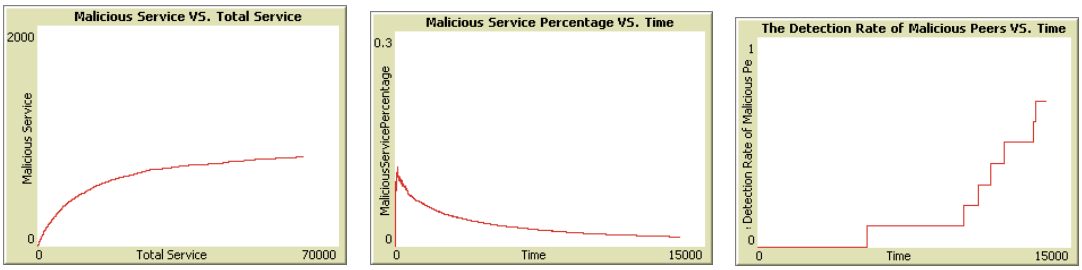
Notation	**Environment Parameter**	**Default Value**
N	Network size	100,500
α	Malicious peer percentage	10%,40%
μ_r	Initial trust rating distribution mean value	25,75
σ_r^2	Initial trust rating distribution variance	10
μ_c	Initial credibility factor distribution mean value	5
σ_c^2	Initial trust rating distribution variance	1
θ_v	Peer voting threshold	50%

credibility peers, and others. The simulated network is in the central panel, showing the P2P Desktop Grid network behavior. In the panel, the green shapes represent the normal peers and red ones the malicious peers. The circles represent peers with normal credits, while the triangles are peers with very low credibility that always make dishonest recommendations. On the right side, it lists three plotting panels that illustrate results as simulations evolve.

The basic parameter setting and default values used in the simulation are summarized in Table 6.3. The network is configured as 100 nodes and 500 nodes, with malicious peer percentage (α) varying from 10% to 40%. For honest peers, the initial trust value follows a normal distribution with mean $\mu_r = 75$ and variance $\sigma_r^2 = 10$. For malicious peers, initial trust value follows a normal distribution with mean $\mu_r = 25$ and variance $\sigma_r^2 = 10$. The initial credibility factor value follows a normal distribution with mean $\mu_c = 7.5$ and variance $\sigma_c^2 = 1$. We use a voting threshold of $\theta_v = 50\%$

which means one peer is identified malicious if more than 50% of eligible peers vote it as malicious.

The primary metrics we use to assess the performance of the reputation system are "the percentage of malicious services" and "The detection rate of malicious peers". The initial experiments are done simulating a network with 100 peers where 10% of the peers are malicious and 10% of the peers have bad credibility. The simulation of a P2P grid network proceeds in simulation cycles: each simulation cycle is subdivided into a number of service distribution cycles. In each service cycle, a peer in the network may be actively submitting a job. When distributing a job, a peer waits for incoming responses, selects a most reputable peer to distribute the job. When a job cycle completes, the peer inserts a new transaction record to its History Table.



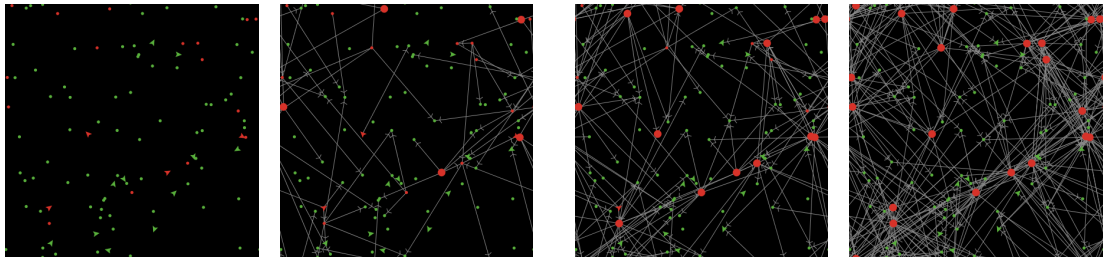
(a) Malicious Service vs. Total Service (b) Malicious Service Percentage vs. Time (c) The Detection Rate of Malicious Peers

Figure 6.2: 100 Nodes Simulation (Malicious Peer Percentage=10%)

The simulation results are shown in Figure 6.2. The computed trust ratings accurately reflect each peer's actual behavior. The number of malicious services starts to decrease with time.

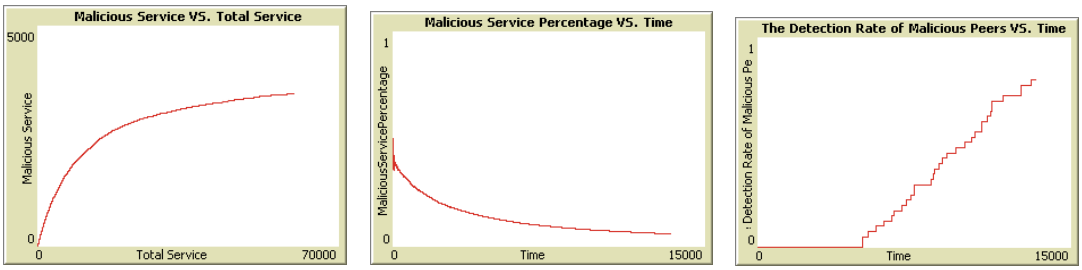
As shown in Figure 6.2 (c), as time goes by, peers in the network gradually isolate malicious peers by votes and finally identify malicious peers. This procedure is also shown in Figure 6.3 (Red dots represent malicious peers. Big dots indicate peers that are identified as malicious peers).

In the second set of experiments, we keep the network size as 100 nodes where 10%



(a) (b) (c) (d)

Figure 6.3: Malicious Peers are Identified



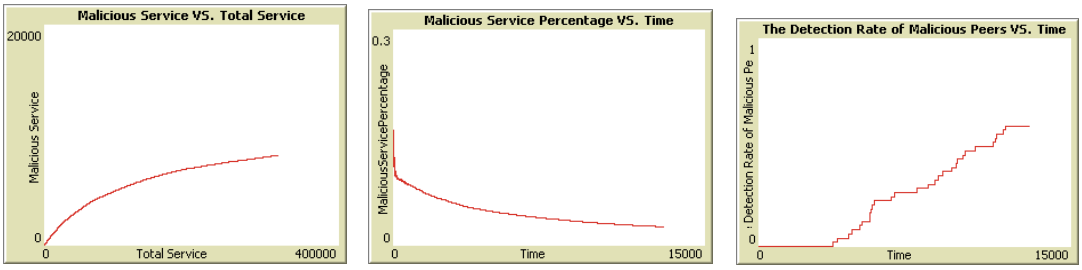
(a) Malicious Service vs. Total Service (b) Malicious Service Percentage vs. Time (c) The Detection Rate of Malicious Peers vs. Time

Figure 6.4: 100 Nodes Simulation (Malicious Peer Percentage=40%)

of the peers have bad credibility. However, the percentage of malicious peers increases up to 40% which means there are lots of more malicious peers in the network. In Figure 6.4, the simulation results demonstrate that the system functions well in the network when a large part of nodes are malicious.

To test the H-Trust performance in large-scale networks, we increase the network size to 500 peers in the third set of experiments. The percentage of malicious peers is set as 10% and 40% as we did in previous experiment. The simulation results are shown in Figure 6.5 and Figure 6.6.

We observe that in a larger network, H-Trust still functions well and the convergence time increases in a reasonable range duo to more iteration complexity caused by larger-scale networks. Both results demonstrate that the H-Trust system detects most of the malicious peers after aggregation iterations. On average, the peers transmit modest messages and incur acceptable communication overheads.

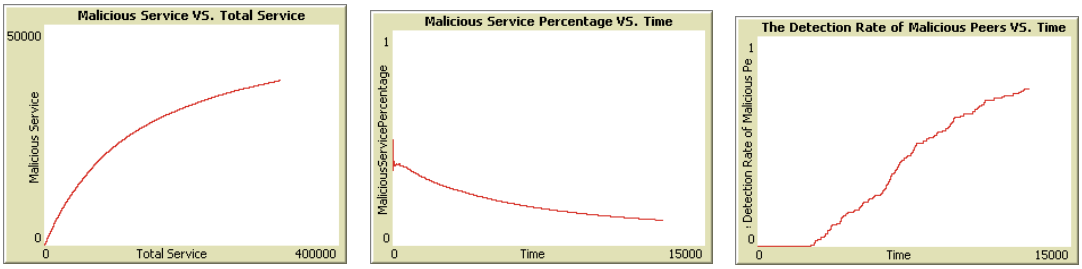


(a) Malicious Service vs. Total Service

(b) Malicious Service Percentage vs. Time

(c) The Detection Rate of Malicious Peers

Figure 6.5: 500 Nodes Simulation (Malicious Peer Percentage=10%)



(a) Malicious Service vs. Total Service

(b) Malicious Service Percentage vs. Time

(c) The Detection Rate of Malicious Peers

Figure 6.6: 500 Nodes Simulation (Malicious Peer Percentage=40%)

6.2 Simulate VectorTrust Dissemination in P2P Networks

We evaluate the performance of VectorTrust through simulation. We designed an event-driven simulator prototype VTSim using NetLogo, a popular multi-agent simulation engine [67]. We start with description of simulation setup and primary procedures. Performance metrics are then defined and results in terms of each metric are presented and discussed. At the end of this section, we present a thorough summary of the existing trust and reputation schemes.

6.2.1 Simulation Setup and Procedure

The basic parameter setting and default values used in the simulation are summarized in Table 6.3. The network is configured from 100 nodes to 1000 nodes, with malicious

Table 6.2: Simulation Parameter Setting (VectorTrust)

Notation	**Environment Parameter**	**Default Value**
N	Network size	100,200 to 1000(step by 200)
D	Average peer outdegree	3,6,9
α	Malicious peer percentage	10%,30%,50%
μ_r	Initial trust rating distribution mean value	0.25,0.75
σ_r^2	Initial trust rating distribution variance	0.1
μ_D	Peer outdegree distribution mean value	3,6,9 (D)
σ_D^2	Peer outdegree variance	1
γ	Trust rating variance threshold	0.02 to 0.20
λ	Expected number of new transaction occurrences in unit service interval	10 to 50(step by 10)
θ_v	Peer voting threshold	50%
ϵ	ϵ-convergence threshold	0.02

peer percentage (α) varying from 10% to 50%. For honest peers, the initial trust value follows a normal distribution with mean $\mu_r = 0.75$ and variance $\sigma_r^2 = 0.1$. For malicious peers, initial trust value follows a normal distribution with mean $\mu_r = 0.25$ and variance $\sigma_r^2 = 0.1$. The initial trust relationship (in terms of trust link in trust graph) follows random distribution as shown in Figure 6.7. The trust network complexity is represented in terms of nodes' trust outdegree D. A network complexity with $D = 6$ indicates that the initial nodes' trust outdegrees follow a normal distribution with mean $\mu_D = 6$ and variance $\sigma_D^2 = 1$. Figure 6.8 shows network trust density with network trust complexity D=3, 6, and 9 (red points represent malicious peers). Note

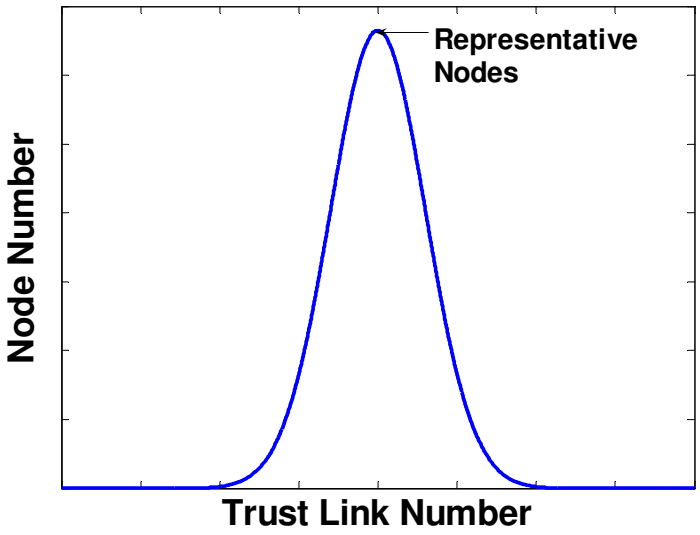
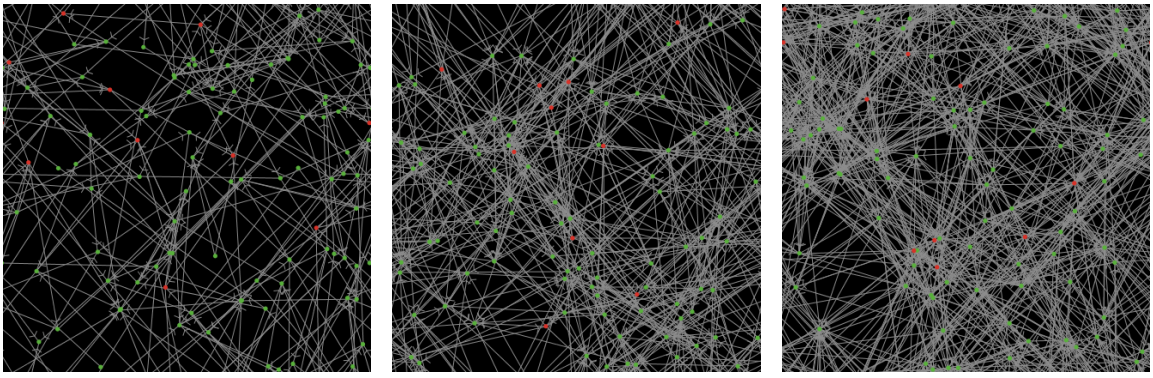


Figure 6.7: Trust Links Distribution (Only distribution function is shown here, detail parameter setting is summarized in Table 6.3)

that, the links in Figure 6.8 shows the initial direct trust relation between peers. In malicious peer detection experiment, we use a voting threshold of $\theta_v = 50\%$ which means one peer is identified malicious if more than 50% of eligible peers vote it as malicious. And for a candidate peer A, the eligible voting peers is defined as the peer set that has either direct or indirect trust rating towards peer A.

The simulation starts with initial parameters as in Table 6.3, generates network topology, and initializes local trust tables following the given distribution. Then the VectorTrust aggregation processes are simulated step by step for all peers concurrently. To measure the performance under dynamic models, new transactions is continuously generated according to a poisson distribution with an arrival rate $\lambda = 10$ to 50 transactions per service cycle, between a random source node and a random destination node. New peer will randomly join the network, and peer leave/die also randomly happens. In such a dynamic model and even in a real P2P network, it is hard to achieve strictly convergence status. In our simulation, the convergence in a dynamic environment is ϵ-convergence, and ϵ-convergence is defined as that the variance between any peer's two consecutive trust tables is smaller than the pre-set



(a) $D = 3$ (b) $D = 6$ (c) $D = 9$

Figure 6.8: Network Complexity Setting. The trust network complexity is represented in terms of nodes' trust outdegree D. The links shows the initial direct trust relation between peers. Green points represent normal/good peers, and red points represent malicious peers.

threshold ϵ.

6.2.2 Results and Analysis

Convergence Time. VectorTrust convergence time is measured in terms of the number of iterations needed to establish each peer's trust table and achieve convergence status. The term "convergence" in a dynamic environment is ϵ-convergence which indicates that the variance between any peer's two consecutive trust table is smaller than the pre-set threshold ϵ.

The relationship between the network size, network complexity and convergence time are shown in Figure 6.9. The figure shows that VectorTrust only needs a small number of aggregation cycles before convergence. For instance, in a 800-nodes P2P network with network complexity $D = 6$, convergence takes only 8 iterations. We also observe that convergence time decreases as network complexity increases. This is reasonable because in a more complex network, each peer has higher connectivity and can obtain more information in one iteration. In a network with N nodes, a

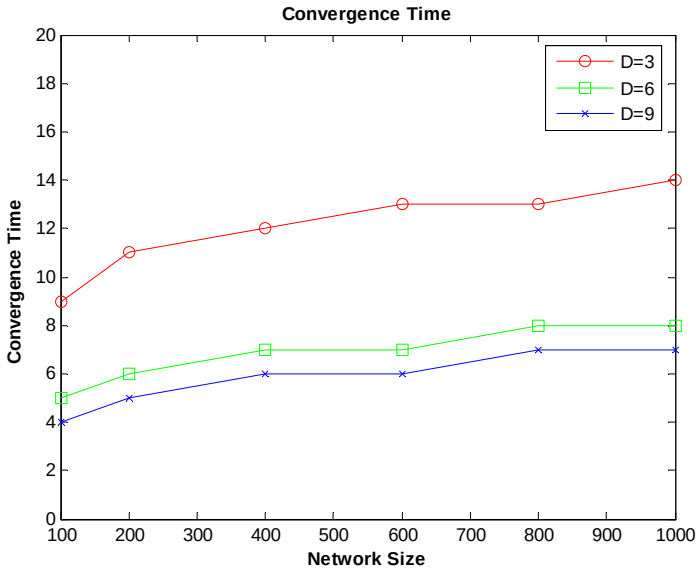


Figure 6.9: Convergence Time

peer needs $\log_D N$ iterations to retrieve all relevant trust information on average. Therefore, as network size N increases, convergence time increases relatively slowly as shown in Figure 6.9, in the order of $O(\log_D N)$. This shows that VectorTrust features satisfactory scalability.

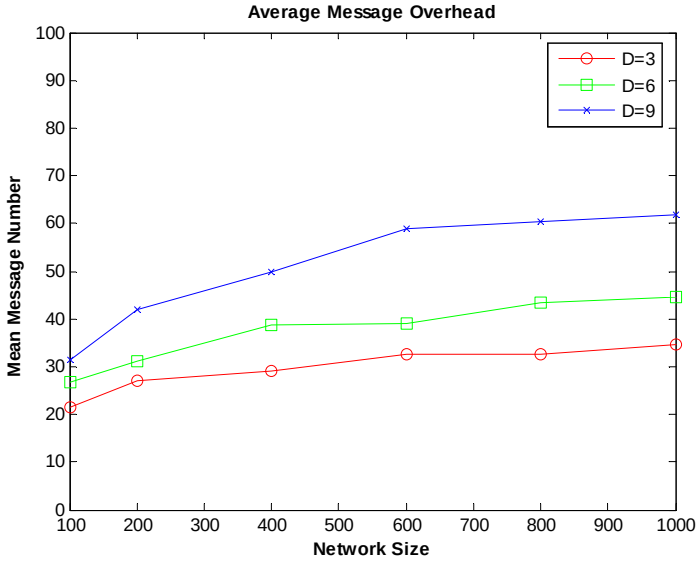


Figure 6.10: Average Message Overhead

Communication Message Overhead. In most of the trust and reputation

schemes, reputation aggregation is quite expensive when the network grows to large number of nodes. Figure 6.18 shows the average communication message overhead for per individual peer to achieve convergence in VectorTrust. The message overhead grows slowly with network size growing, which shows that VectorTrust is a lightweight scheme. The computed trust values do not change significantly after convergence status even if there are new transactions coming. The observation is that in a network with high complexity, VectorTrust system incurs more message overheads but takes less convergence time. Overall, per service cycle, average messages needed for each peer is in the order of $O(D)$ where D is network complexity. In a typical VectorTrust network, D is in the order of $O(\lg N)$. To achieve convergence, it takes $O(\lg N)$ iterations. Therefore, the average message overhead for each peer to converge is $O(\lg^2 N)$. And the average message overhead for each peer in one iteration is $O(\lg N)$.

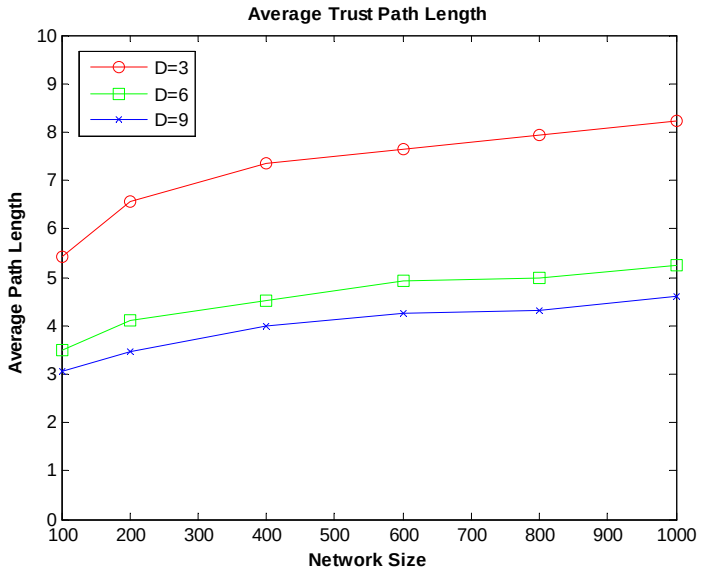


Figure 6.11: Average Trust Path Length

Average Trust Path Length. Average trust path length is a unique evaluation metric for VectorTrust scheme. It indicates the average length of a trust path starts from a source peer to a destination peer in convergence status. The effects of various network size and network complexity on average trust path length are shown in

Figure 6.19. Generally, a peer should reach its target within 6 hops. In most cases, the converged trust path is between 3 and 6. In the network where the complexity is too low, e.g., $D = 3$, lots of trust paths length is more than 6, and the average even reaches 8 hops. This kinds of longer trust paths involve more trust transfers and it leads to lower accuracy in inferred trust ratings.

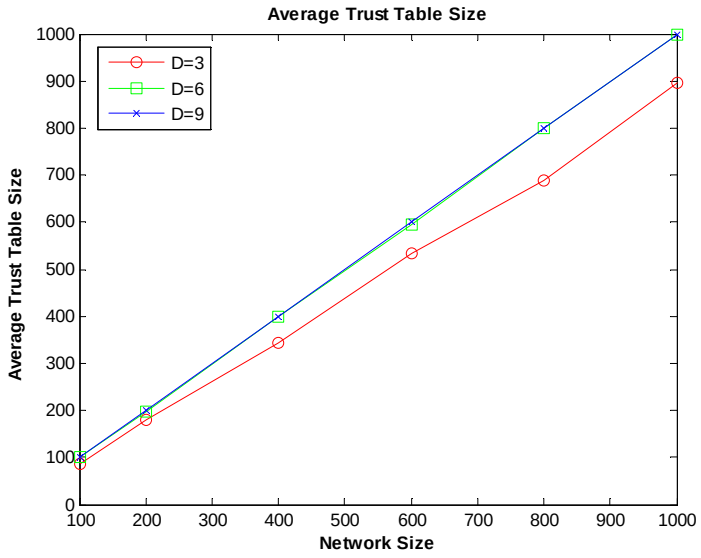


Figure 6.12: Average Trust Table Size

Average Trust Table Size. The average trust table size for each individual peer in VectorTrust is shown in Figure 6.12. The curves correspond to various network sizes and complexities. Trust table size grows with network size. That is, most peers have accesses to all other peers' trust information over the whole network. This fact indicates the trust information spreads fast and extensively in VectorTrust. In a large P2P network, the trust table can have hundreds or thousands of entries. The storage overhead could be a burden. However, the VectorTrust environment is composed mostly by fixed desktops or laptops. The storage requirement for a thousands of entities text trust table is affordable in most terminals. To avoid huge trust table size in a large network of storage limited devices such as PDAs or sensors, we suggest to use group based hybrid trust aggregation. In group based VectorTrust scheme,

multiple peers can be reflected as one entity in a hybrid trust table, which greatly reduces the trust table size. Nodes within the same work group share the group's trust rating. In addition, we can apply Least Recent Used (LRU) mechanism to remove inactive peers/entries in the table.

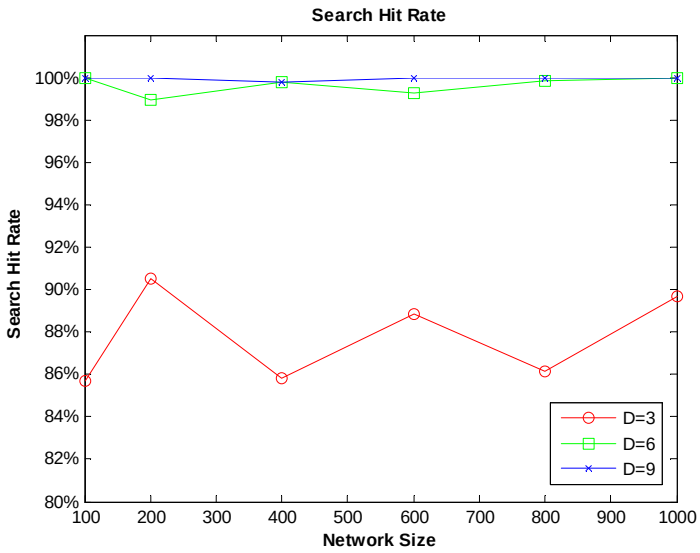


Figure 6.13: Query Hit Rate

Query Hit Rate. A basic function in VectorTrust is trust rating query where one local peer searches in its local trust table trying to find remote peer's trust rating. The query hit rate is the percentage of successfully completed queries over total number of queries issued. As shown in Figure 6.13, when the network complexity exceeds a special threshold. The query hit rate approaches 100%. In a less complex network where peers have only a few connections to other peers, the query hit rate is much lower. In reality, such low network complexity does not happen often. We test this kind of less complex network in experiment only for comparison purpose. Note that, 100% hit rate does not indicate 100% trust ratings in local trust tables correctly reflect the trust features of remote peers. The correctness of local trust tables is discussed below.

Accuracy. VectorTrust aggregation accuracy is measured by "Effective Trust

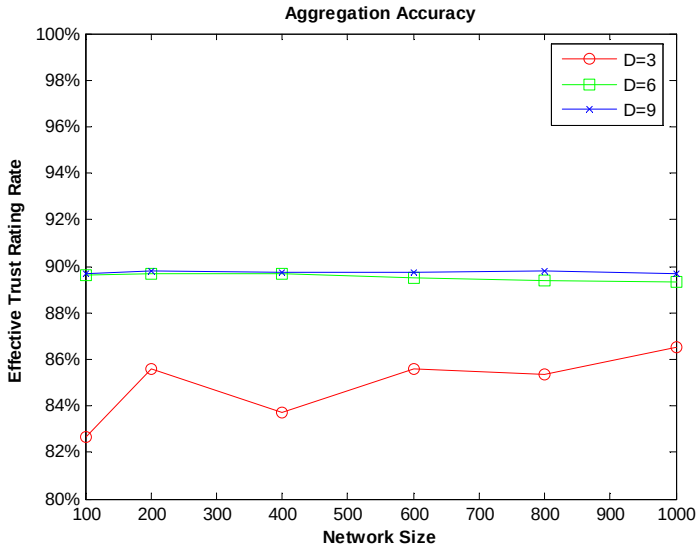
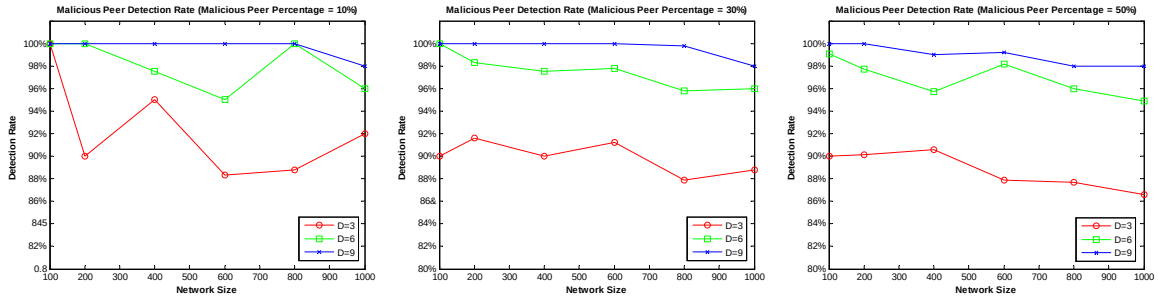


Figure 6.14: Aggregation Accuracy

Rating Rate" shown by Figure 6.14. One inferred trust rating is effective if and only if the variance between the trust rating and remote peer's real behavior (represented by a pre-set rating score) is less than the pre-set threshold γ. On average, the accuracy of VectorTrust is maintained around 90%. This result is very encouraging because VectorTrust is a personalized trust system using inferred (not direct) trust and the information for each node to access is limited. When the network complexity is low, the aggregation accuracy decreases a lot. As we discussed before, in less complex network, the longer trust paths involves more trust transfers, which leads to lower accuracy in inferred trust ratings. To eliminate the effect of long trust paths, we advise to use weighted trust rating $r_w = \sqrt[d]{r}$ (r: trust rating, d: total hops).

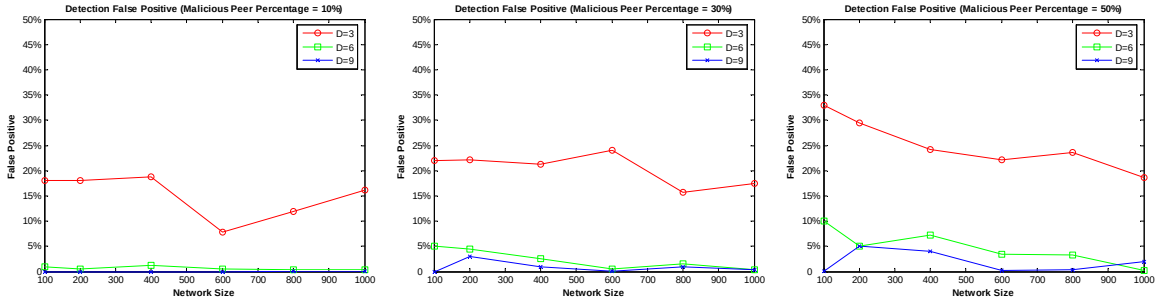
Malicious Peer Detection. The malicious peer detection rate and detection speed are measured. We study the problem of inferring indirect trust when direct trust information is not available. So the malicious behavior in trust rating generation period is beyond the scope of this paper (which was discussed in [11]). In this paper, we study the malicious behavior in trust path aggregation and trust transferring process. Malicious peers cheat in transactions and it sends fake trust tables to peers



(a) Malicious Peer Detection Rate ($\alpha = 10\%$)

(b) Malicious Peer Detection Rate ($\alpha = 30\%$)

(c) Malicious Peer Detection Rate ($\alpha = 50\%$)



(d) Detection False Positive ($\alpha = 10\%$)

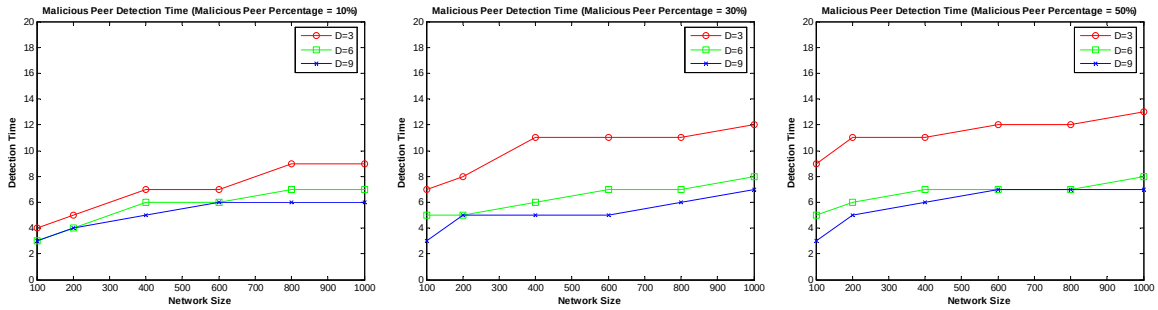
(e) Detection False Positive ($\alpha = 30\%$)

(f) Detection False Positive ($\alpha = 50\%$)

Figure 6.15: Malicious Peer Detection Rate

they interact with. In this simulation, the malicious peers adopt a mix of false and true information strategy. The fake message rate is set as %50 which indicates half of the tables a malicious peer sent is fake. In our simulation, peers start voting phase periodically. One peer is identified as malicious if more than θ_v eligible peers in the network vote it as malicious. As claimed, for a candidate peer A, the eligible voting peers is defined as the peer set that has either direct or indirect trust rating towards peer A. Initial malicious peer percentage is set as $\alpha = 10\%, 30\%, 50\%$.

Figure 6.15 reports the malicious peer detection rate and the related false positive rate as a function of network size. In general, the false positive rate increases reversely proportional to the detection rate. In a normal network setting ($D = 6, 9$, $\alpha = 10\%, 30\%$), more than 95% of malicious peers are identified with a false positive rate less than 5%. Even in a less complex network ($D = 3$), the detection rate still achieves



(a) $\alpha = 10\%$ (b) $\alpha = 30\%$ (c) $\alpha = 50\%$

Figure 6.16: Malicious Peer Detection Time

90%. But in such a network, the false positive is rather high.

VectorTrust detection malicious peer speed is shown in Figure 6.16 under different malicious percentage configurations. Increasing the percentage of malicious peers does not lead to lower performance. VectorTrust is able to detect most of malicious peers before aggregation converges. And malicious peers are much easier to be detected in a complex network. Given a network with 1000-nodes with $D = 6$ and $\alpha = 30\%$, for example, VectorTrust detects more than 95% malicious peers in 8 iterations.

6.3 Simulate cTrust in Cyclic Mobile Environment

6.3.1 Experiment Setup

CMANET Contact Pattern Model. We construct an unstructured network based on the NUS student trace. The data of contact patterns is from the class schedules for the Spring semester of 2006 in National University of Singapore (NUS) among 22341 students with 4875 sessions [68, 69]. For each enrollment student, we have her/his class schedule. It gives us extremely accurate information about the contact patterns among students over large time scales. The contact patterns among students inside classrooms were presented in [68]. Following class schedules, students move around on campus and meet each other when they are in the same session. The trace data set considers only students movements during business hours, and ignores contacts

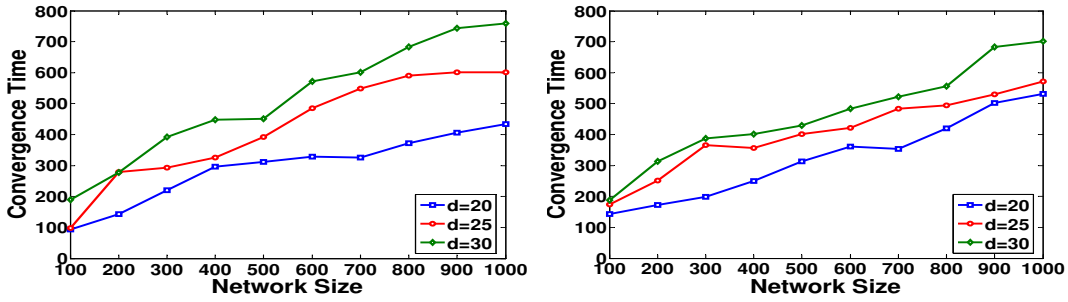
that students hang around campus for various activities outside of class. The time is compressed by removing any idle time slots without any active sessions. So the contacts take place only in classrooms. Two students are within communication range of each other if and only if they are in the same classroom at the same time. The sessions can be considered as classes. The unit time is one hour, and a session may last multiple hours. The NUS contact patterns can be modeled as CMANET. In our experiment, 100 to 1000 students are randomly chosen to simulate 100 to 1000 moving peers in CMANET. Following her/his class schedule, each student appears moving cyclically in classrooms. The contact probability P is set as 0.9 which indicates that when two nodes meet, they have a probability of 0.9 to communicate. We considered all 4875 sessions in the data set. The time for the whole system cycle (C_S) is 77 hours (time units).

Trust Topology Model. The random trust topology and scale-free trust topology are used to establish trust relationships in this simulation. In random trust topology, the trust outdegree of a peer follows normal distribution with mean value $\mu_d = 20, 25, 30$ and variance $\sigma_d^2 = 5$. On the random trust topology, all peers have similar initial trust links. Under the scale-free trust topology, highly active peers possess large numbers of trust links, and most other peers only have small numbers of trust links. The number of trust links follows power law distribution with an scaling exponent $k = 2$.

Parameter Setting. The network is configured from 100 nodes to 1000 nodes. The network complexity is represented in terms of nodes' average outdegree d. A network complexity with $d = 20, 25, 30$ indicates on average, the initial nodes' outdegrees is $20, 25, 30$. Peer's real behavior is represented by a pre-set normal distribution ($\mu_r = 0.25, 0.75$, $\sigma_r^2 = 0.2$) rating score $r \in [0, 1]$. As mentioned in Section **??**, we assume the accurate direct trust ratings have already been generated. This is reasonable because any trust inference scheme must rely on an accurate trust rating

scheme. It is meaningless to study the inference trust based on the direct trust rating if the direct trust rating is not reliable. So in our simulation, the direct trust rating $R \in [0, 1]$ is generated with a normal distribution ($\mu_R = r$, $\sigma_R^2 = 0.1$) based on peer's real behavior score r. The parameters in iteration function is set up as learning rate $\alpha = 1$, discount factor $\gamma = 1$ and adjusting factor $n_a = 9$. To measure the performance under dynamic models, new transactions is continuously generated according to a poisson distribution with an arrival rate $\lambda = 10$ to 50 transactions per service cycle, between a random source node and a random destination node. New node will randomly join the network, and peer leave/die also randomly happens. In such a dynamic model and in a real mobile ad hoc network, it is hard to achieve strictly convergence status. So the convergence in our simulation is ϵ-convergence, and ϵ-convergence is defined as that the variance between any peer's two consecutive trust tables is smaller than the pre-set threshold $\epsilon = 0.02$.

6.3.2 Results and Analysis

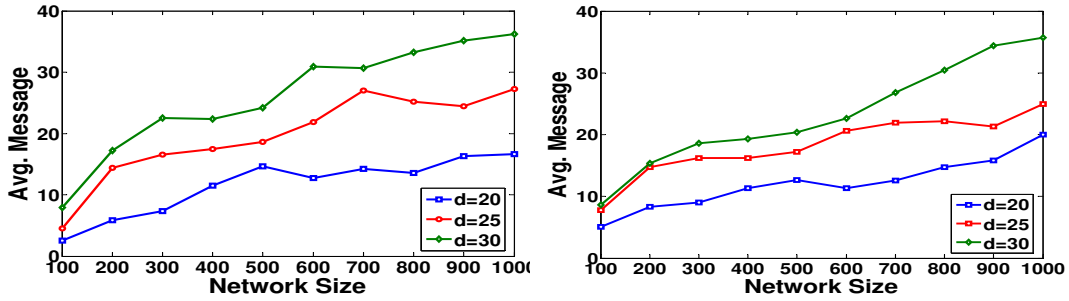


(a) Random Trust Topology (b) Scale-Free Trust Topology

Figure 6.17: Convergence Time

Convergence Time. The convergence time is measured in terms of the number of time units needed to achieve ϵ-convergence status. Figure 6.17 shows that cTrust only needs a small number of aggregation cycles before convergence. We also observe that convergence time increases as network complexity increases. As network size N increases, convergence time increases relatively slowly ($O(n)$). This shows that

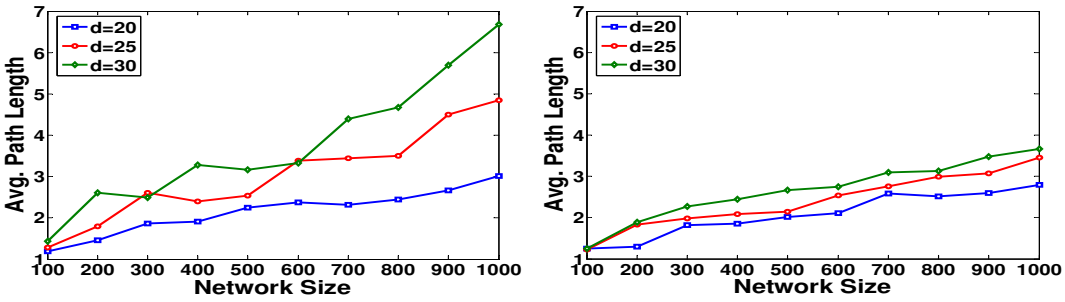
cTrust features satisfactory scalability. We also observe that the trust topologies do not affect the convergence time so much as network size.



(a) Random Trust Topology (b) Scale-Free Trust Topology

Figure 6.18: Message Overhead

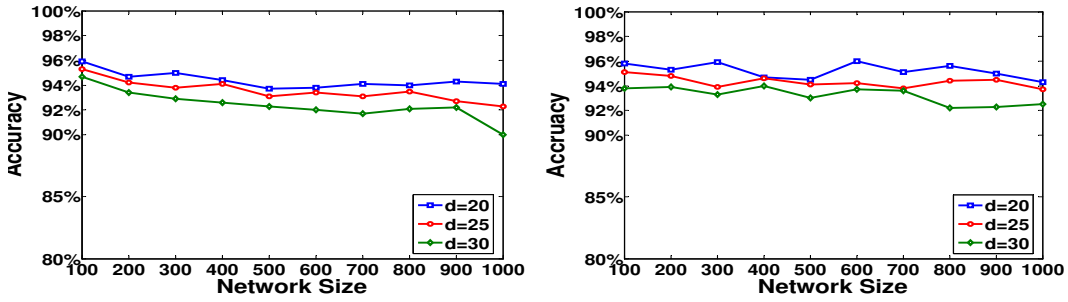
Communication Message Overhead. Figure 6.18 shows the average communication message overhead to achieve convergence per individual peer. cTrust greatly reduce the communication message overhead by using MDP model. This is because, in each iteration, each node only receive trust table for one of its most trusted neighbors (Equation (4.5)). The message overhead grows slowly as the network size grows, showing that cTrust is a lightweight scheme. In a network with high complexity, cTrust system incurs more message overheads. In a typical cTrust network, the average message overhead is affected by only network size N and complexity d and not affected by trust topology. As a result, the overhead curves for both topologies in the figures appear similar.



(a) Random Trust Topology (b) Scale-Free Trust Topology

Figure 6.19: Average Trust Path Length

Average Trust Path Length. Figure 6.19 indicates the average length of a trust path starts from a source peer to a destination peer in convergence status. Generally, the trust path length increases with the network size and complexity, which indicates peers gain more remote trust information. In the scale-free trust topology, the trust path length is greatly reduced. This is because in scale-free trust topology most peers have only a few connections while some power peers control many links, making trust information hard to spread. In a complex network where trust information can be spread father, there are more longer trust paths and involve more trust transfers.



(a) Random Trust Topology (b) Scale-Free Trust Topology

Figure 6.20: Aggregation Accuracy

Accuracy. cTrust aggregation accuracy is measured by comparing all the inferred trust ratings with peers real behavior scores. The similarity is considered as aggregation accuracy. As shown in Figure 6.20, on average, cTrust aggregation accuracy is maintained above 90%. The result is very encouraging because cTrust is a personalized trust system using inferred (not direct) trust and the information for each node to access is limited in CMANETs. As the network complexity increases, the accuracy decreases. This is because in complex networks, there are more long trust paths that involve more trust transfers, resulting in lower accuracy in inferred trust ratings due to multi-hop relationships. The accuracy in scale-free trust topology is slightly higher than in random trust topology. One reason is in scale-free trust topology, the average trust path is shorter which leads to high accuracy in trust transfer.

6.4 Incentive Mechanism Experimental Evaluation

6.4.1 Incentive Mechanism Simulation Setup

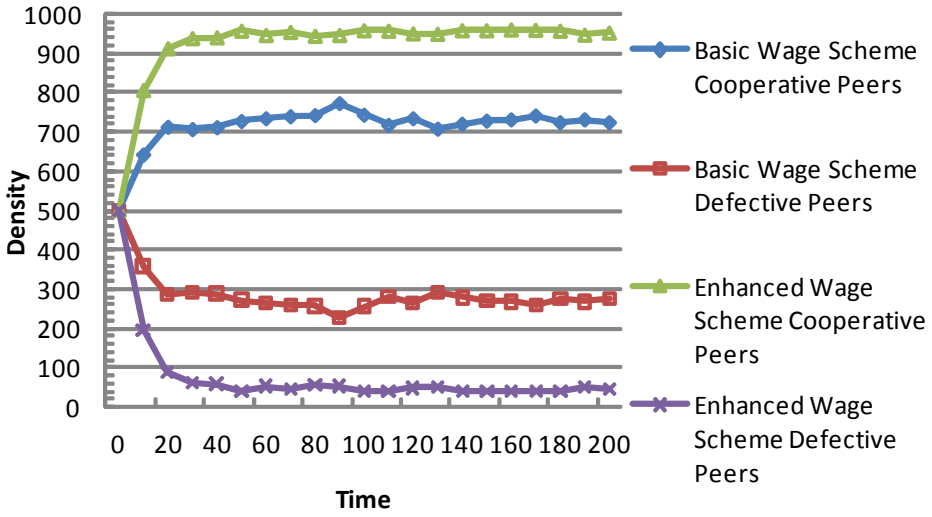


Figure 6.21: The Density of Strategies

A P2P network for file sharing is simulated. The basic parameter setting and default values used in the simulation are summarized in Table 6.3. The network consists of 1000 nodes with an average node lifetime of $T = 200$ hours. All nodes are independently and identically distributed in the system. To evaluate the performance in dynamic environment, new generated nodes are continuously joining the network according to a poisson distribution (arrival rate $\lambda = 10$ per iteration cycle), and nodes die and nodes leave are also randomly happened. The node population size stays constant which is 1000 in our experiment. The storage space of each node is $Mem = 1$MB. We assume there are small files stored per nodes. The unit time is set up as 1 hour which represents one feedback iterations round. Each peer was assigned an initial currency/credit following a normal distribution with mean $\mu_c = 50$ and variance $\sigma_c^2 = 10$. We assume the utility can be negative. The file download transactions occur in each iteration, with the occurrence rate $\beta = 10\%$ of the total population. In each iteration, two nodes are randomly paired to start small file

download transaction once. Mutual transactions are reported by reporters. A third node acts as the querist and it queries the transaction type.

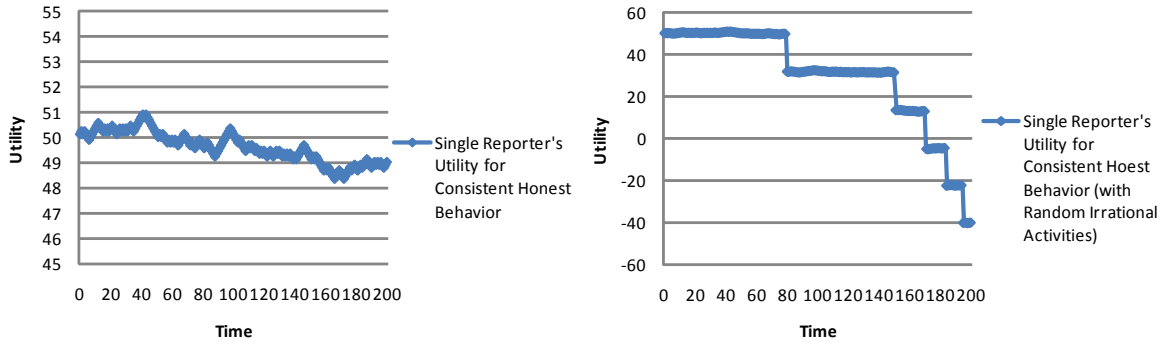
We use the environment parameter setting in Section **??** as the "basic wage scheme" where $\alpha_s = 3$, $\alpha_f = 1$ and $P = 50\%$. Accordingly, the outputs are $q_f = 0.5$, $q_s = 4.5$, and the wages are $w_s \approx 2.36$, $w_f \approx 0.14$. We assume the P2P system is a homogeneous system. All transactions consume the same cost to reporters and provide the same output to querists. To compare the performance, we also propose the "enhanced wage scheme" in which we increase wage to $w_s \approx 2.5$ to eliminate $\{m = failed, T = successful\}$ equilibrium. In the enhanced wage scheme, w_f stays the same as in the basic wage scheme where $w_f \approx 0.14$.

All nodes are self-interested, and they can change their interaction strategies/behavior adaptively to maximize their own utility (credits in this experiment). A cooperative node indicates that a node provides truthful feedback, while a noncooperative node represents a node that gives dishonest report. For each reporting round, the reporter should evaluate the cost and the expected wage rationally to determine each message feedback submission. However, to measure some unpredictable behavior in real-world application, we allow peer to randomly adopt some irrational activities at a low probability 5%. At the start stage, there are a mix of 500 cooperative nodes and 500 noncooperative nodes.

The simulation generates P2P network topology, and initializes peers' initial credit/currency with the mean value 50. The process of the incentive scheme is simulated step by step for all peers concurrently.

6.4.2 Incentive Mechanism Results and Analysis

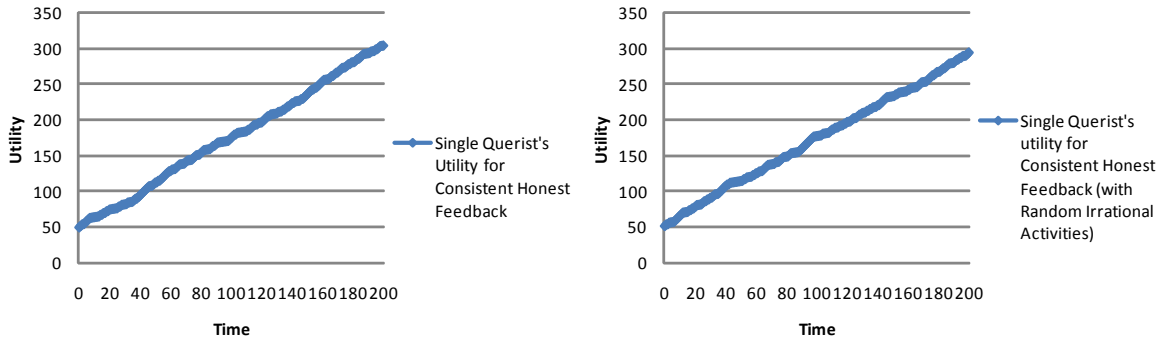
We define the density of a strategy as the number of the nodes employing the strategy (cooperative/noncooperative) among all the nodes. The relation between strategies density and time in basic wage scheme and enhanced wage scheme is shown in Fig-



(a) Utility for Consistent Honest Behavior

(b) Utility for Consistent Honest Behavior (with Random irrational Activities)

Figure 6.22: Single Reporter's Utility for Honest Behavior

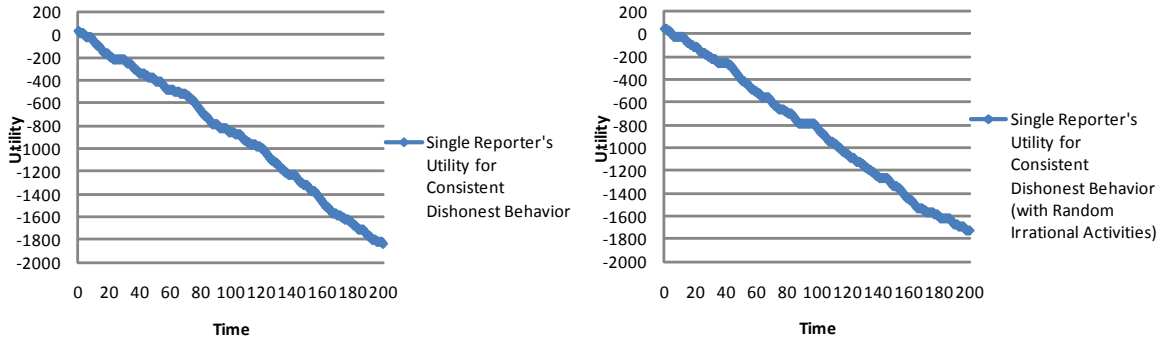


(a) Utility for Consistent Honest Feedbacks

(b) Utility for Consistent Honest Feedbacks (with Random irrational Activities)

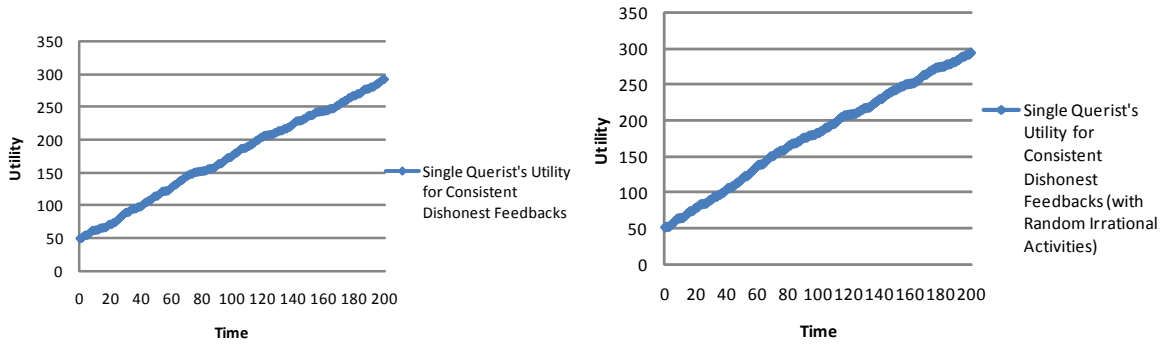
Figure 6.23: Single Querist's Utility for Honest Feedbacks

ure 6.21. At the starting stage, there are 500 cooperative nodes and 500 noncooperative nodes. After a few iterations, the cooperative nodes dominate the population of the network in both the two schemes. For basic wage scheme, the cooperative population is around three fourths which is less than the population in enhanced wage scheme. This is because in basic wage scheme, when the transaction state is "successful" there is no incentive for peers to report truth. That is, in the "successful" status, basic wage scheme has two equilibrium $\{m = successful, T = successful\}$ and $\{m = failed, T = successful\}$, which results a portion of dishonest report. In enhanced wage scheme, the cooperative nodes increase sharply for the reason that the enhanced wage eliminate the $\{m = failed, T = successful\}$ equilibrium. Nodes



(a) Utility for Consistent Dishonest Behavior (b) Utility for Consistent Dishonest Behavior (with Random irrational Activities)

Figure 6.24: Single Repoeter's Utility for Dishonest Behavior

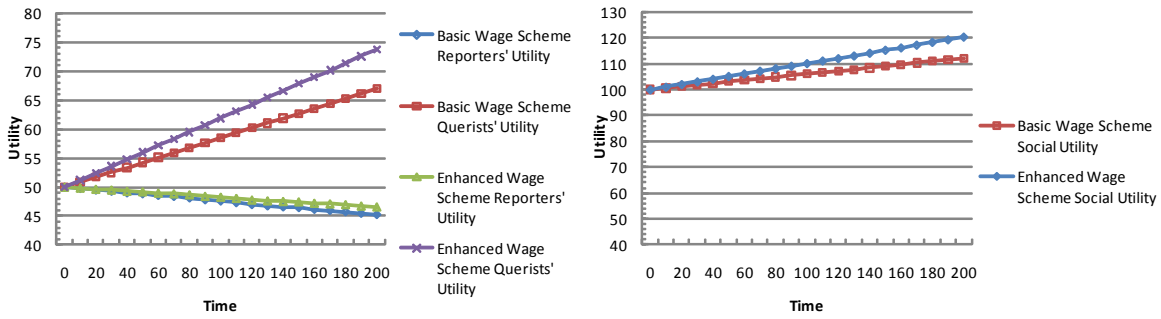


(a) Utility for Consistent Dishonest Feedbacks (b) Utility for Consistent Dishonest Feedbacks (with Random irrational Activities)

Figure 6.25: Single Querist's Utility for Dishonest Feedbacks

receive higher payoff if they report truth in "successful" state. The cooperative peers cannot achieve 100% because we allow peers to randomly adopt some irrational activities at a low probability 5%, which leads to some unpredictable reporting activities.

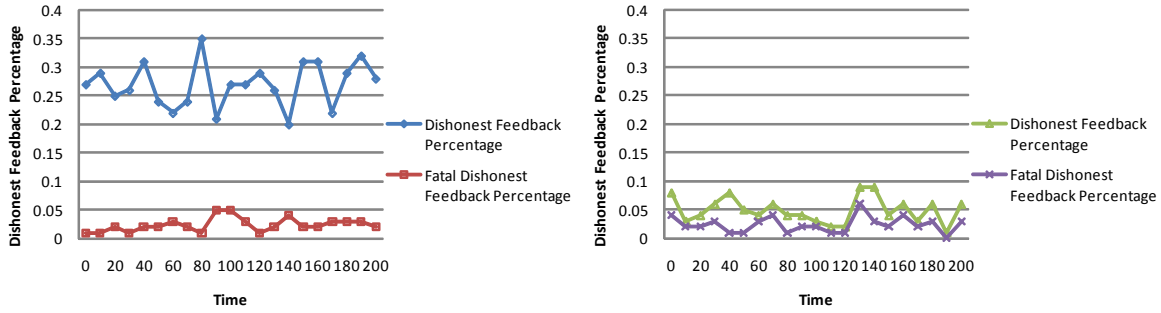
In the basic wage scheme, assuming a reporter always adopts honest feedbacks, its utility is shown in Figure 6.22(a). At time 0, The initial average utility is set up as 50. The results show that the reporter's utility keeps the same around 50 with time going on. This is consistent with the analytical results calculated by Formula (5.18) (the reporters' average utility maintains at the same level). Figure 6.22(b) shows the results when the reporter is allowing to randomly adopt some irrational behavior. In this case, the random behavior may lead to dishonest feedbacks. And as



(a) Querists' and Reporters' Average Utility V.S. Time

(b) Average Social Utility V.S. Time

Figure 6.26: Average Utility for Reporters, Querists and System



(a) Basic Wage Scheme

(b) Enhanced Wage Scheme

Figure 6.27: Dishonest Feedback Rate

the curve shown, node's utility decreases every time when it submits dishonest feedback. Accordingly, the single querist's utility is shown by Figure 6.23 when reporter's feedbacks are honest. In both cases, querist's utility keeps increasing. The curves in Figure 6.23(a) and 6.23(b) demonstrate that the reporter's irrational behavior in honest reporting does not significantly affect querist's utility.

In basic wage scheme, when a reporter always gives dishonest feedbacks, its utility is shown by Figure 6.24(a). The reporter's utility keeps decreasing sharply because every $\{m = succcessful, T = failed\}$ feedback will lead to a loss 17.89 units of utility for the reporter as shown in Figure **??**. When the reporter is allowed to randomly adopt some irrational behavior, its utility is slightly higher (shown by Figure 6.24(b)) than consistent dishonest feedbacks. And the random behavior may lead to a small portion of honest feedbacks to gain some positive utility. Corresponding to

the reporter's dishonest behavior, the single querist's utility is shown by Figure 6.25. Similarly, the curves in Figure 6.25(a) and 6.25(b) prove that the reporter's irrational behavior in dishonest reporting does not significantly affect querist's utility. Comparing the results in Figure 6.23 and Figure 6.25, we could notice that querist's utility will not be significantly affected by reporter's feedback strategies.

Figure 6.26(a) shows the querists' and reporters' average utility in basic wage and enhanced wage scheme. Querists' average utility keeps increasing linearly in both of the schemes. In enhanced wage scheme, querists' utility increase faster than the basic scheme. This is because the enhanced wage eliminate $m = failed$ report at $T = successful$ status, it leads to more output credits for querists. The reporters' average utility keeps decreasing which is a little different from the analytical results calculated by Formula (5.18) (the reporters' average utility maintains at the same level) because we allow peers to randomly adopt some irrational behavior at a low probability. Reporters employ irrational behavior with a small probability (5%). The reporters' average utility is higher in enhanced wage scheme than basic scheme for the reason that querists provide reporters more wage in enhanced wage scheme. The querists achieve maximum utility while enforcing the truthful feedback when they enhance the wage as little as it can be above the boarding line. The average social utility which is the sum of the querists' utility and reporters' utility is shown by Figure 6.26(b). By slightly enhancing the wage, we achieve higher social utility in enhanced wage scheme.

Dishonest feedback indicates the feedback that is not consistent with the real transaction type. Fatal dishonest feedback is defined as $\{m = succcessful, T = failed\}$, which can damage the reputation system because it hides a failed transaction. Dishonest feedback and fatal dishonest feedback rate in both schemes are shown in Figure 6.27(a) and Figure 6.27(b). Our solution greatly prevents fatal dishonest feedbacks from happening. $\{m = succcessful, T = failed\}$ feedbacks are less than

5% in both schemes. Note that, lots of the dishonest feedbacks are generated by nodes' irrational behavior which are preset in simulation environment parameters. In the basic scheme, there is a large portion of $\{m = failed, T = successful\}$ feedbacks, and it is reduced to less than 10% in enhanced wage scheme. The experimental results show that our scheme reinforces truthful reporting and functions well even if a large portion of transactions are failed ($P = 50\%$).

Table 6.3: Simulation Parameter Setting (WIM)

Notation	**Environment Parameter**	**Default Value**
N	Network size	1000
T	Node lifetime	200 hours
Mem	Node storage space	1MB
P	The probability for a random transaction to be successful	50%
α_s	Cost factor for success transaction	3
α_f	Cost factor for fail transaction	1
w_s	Wage for "successful" feedback	2.36, 2.5
w_f	Wage for "failed" feedback	0.14
q_s	Output for "successful" feedback	4.5
q_f	Output for "failed" feedback	0.5
μ_c	Initial currency distribution mean value	50
σ_c^2	Initial currency distribution variance	10
β	Transaction occurrence rate	10%
P_{act}	irrational activities rate	5%
λ	Expected number of new generated nodes per iteration cycle	10

CHAPTER 7

CONCLUSIONS

We presented the TrustNet scheme to manage trust in peer-to-peer networks, specifically, we proposed the H-Trust aggregation scheme, the trust vector, trust inference and trust vector dissemination algorithms, and cTrust dissemination in cyclic mobile space.

H-Trust is a personalized reactive selective aggregation based group reputation system H-Trust for collaborative resource sharing and execution in P2P desktop grids. Simulation results demonstrated that H-Trust can quickly identify malicious peers under system uncertainties and incomplete information about states of large-scale distributed systems and applications. As a result, H-Trust provides an efficient and robust means to prevent spreading of malicious content. H-Trust allows individual peers to compute local trust values for other users or groups using their own inference algorithm of choice, and thus can be used to implement a variety of distributed and heterogeneous policies.

VectorTrust scheme is proposed to manage trust in peer-to-peer networks with static topology. The experimental results show that VectorTrust scheme is efficient in trust aggregation. After $O(\log_D N)$ iterations, most peers establish local trust tables and reach convergence. VectorTrust convergence time and message overhead increase slowly with network size growing, demonstrating its high scalability. The trust information spreads fast and extensively in VectorTrust. The computational overhead of VectorTrust is reasonable comparing to the existing trust schemes which makes it easy to deploy in the decentralized environment. For example, EigenTrust

is presented as a global trust metric, while VectorTrust as a local trust metric. In a large network with dynamic topology the EigenTrust server has to calculate the eigen value for the whole network from time to time, which involves heavy computational overhead in the server end. In VectorTrust each peer just has to relax trust value by bellman-ford algorithm. The lightweight local relax algorithm makes VectorTrust more suitable to decentralized environment without powerful server. The comparison results also show that VectorTrust scheme performs well even when a large number of peers are malicious.

As expected, the trust rating inference accuracy in VectorTrust scheme is less than a traditional global reputation system. This fact suggests that we still have to improve the aggregation algorithm of VectorTrust and make it more accurate. The slightly lower accuracy is reasonable because of the nature of VectorTrust scheme. VectorTrust uses inferred trust rating in stead of direct experience. VectorTrust aggregates trust ratings with a limited direct experience in the network where most of current schemes require a large number of existing direct experiences to compute trust ratings. VectorTrust is designed to be used in a distributed and decentralized network with no global trust information. Another important observation is that VectorTrust works better in more complex networks. This is encouraging because the current P2P networks are becoming more complex and VectorTrust is suitable to such kinds of networks.

cTrust scheme is aimed to provide a common framework to enable trust inferring in a CMANET trust landscape. We presented the trust transfer function, trust value iteration function, and the cTrust distribution trust aggregation algorithm. To validate our proposed algorithms and protocols, we conducted extensive evaluation based on NUS students trace data. The experimental results demonstrate that cTrust scheme trust aggregation is efficient. cTrust convergence time increases slowly with network size. Message overhead in cTrust is modest. The trust information spreads

fast and extensively in CMANETs. The trust rating inference accuracy in cTrust scheme is over 90%. We believe that cTrust establishes a solid foundation to design trust-enabled applications and middleware in CMANETs.

To gain better understanding of trust landscape in TrustNet, we also introduced the notion of trust spanning tree and trust transitive closure. We have presented the game theoretic model and wage-based incentive mechanism to encourage truthful feedback in reputation systems. Our contributions are multifold. (1) Assuming peers in reputation systems are self-interested, we modeled the feedback reporting problem as the reporting game. (2) we designed a wage-based incentive mechanism for enforcing truthful report. Different from most existing schemes, our algorithm does not require peers to verify the information truthfulness. Leveraging the mechanism design theory, a set of rules including participation constraint and incentive compatibility constraints were defined in detail. The solution requires only localized wage payment schemes. (3) To gain better understanding of landscape in our scheme, initial characteristics of our scheme were investigated.

To validate our proposed algorithms, we conducted extensive simulation-based experiments. The simulation results demonstrate that TrustNet is efficient, accurate, scalable, and robust. We believe that TrustNet establishes a solid foundation to support trust-enabled applications and middleware.

The ongoing and future work includes two aspects. First, we would like extend each trust aggregation/dissemination by considering our proposed incentive mechanisms and peer anonymity and evaluate the performance to ensure fairness, truthfulness and system efficiency. Second, we will deploy TrustNet system in a realworld P2P and ad hoc networks environment to measure its performance with realworld applications.

BIBLIOGRAPHY

[1] S. Androutsellis-Theotokis and D. Spinellis, "A survey of peer-to-peer content distribution technologies," *ACM Comput. Surv.*, vol. 36, no. 4, pp. 335–371, 2004.

[2] I. Stoica, R. Morris, D. Karger, M. F. Kaashoek, and H. Balakrishnan, "Chord: A scalable peer-to-peer lookup service for internet applications," in *ACM SIGCOMM 2001*, (San Deigo, CA), pp. 149–160, August 2001.

[3] S. Ratnasamy, P. Francis, M. Handley, R. Karp, and S. Shenker, "A scalable content-addressable network," in *ACM SIGCOMM 2001*, (San Deigo, CA), August 2001.

[4] D. S. S. Group, "Gnutella: To the bandwidth barrier and beyond,http://www.clip2.com," Nov 2000.

[5] J. Kubiatowicz, D. Bindel, Y. Chen, P. Eaton, D. Geels, R. Gummadi, S. Rhea, H. Weatherspoon, W. Weimer, C. Wells, and B. Zhao, "Oceanstore: An extremely wide-area storage system," tech. rep., U.C. Berkeley Technical Report UCB//CSD-00-1102, 1999.

[6] Bittorrent, "http://www.bittorrent.com/protocol.html."

[7] A. Rowstron and P. Druschel, "Pastry: scalable, decentraized object location and routing for large-scale peer-to-peer systems," in *Proceedings of the 18th IFIP/ACM International Conference on Distributed Systems Platforms (Middleware)*, nov 2001.

[8] skype, "www.skype.com."

[9] P. Resnick, R. Zeckhauser, and a. K. K. E. Friedman, "Reputation systems," *Communications of the ACM*, vol. 43, pp. 45–48, December 2000.

[10] L. Xiong and L. Liu, "Peertrust: supporting reputation-based trust for peer-to-peer electronic communities," *Knowledge and Data Engineering, IEEE Transactions on*, vol. 16, pp. 843–857, July 2004.

[11] S. D. Kamvar, M. T. Schlosser, and H. G.-Molina, "The eigentrust algorithm for reputation management in p2p networks," in *Proceedings of the 12th International Conference on World Wide Web (WWW)*, (Budapest,Hungary), pp. 640–651, May 20-24 2003.

[12] R. Zhou and K. Hwang, "Powertrust: A robust and scalable reputation system for trusted peer-to-peer computing," *IEEE Transactions on Parallel and Distributed Systems*, vol. 18, pp. 460–473, May 2007.

[13] R. Zhou, K. Hwang, and M. Cai, "Gossiptrust for fast reputation aggregation in peer-to-peer networks," *IEEE Transactions on Knowledge and Data Engineering*, vol. 20, no. 9, pp. 1282–1295, 2008.

[14] S. Song, K. Hwang, R. Zhou, and Y.-K. Kwok, "Trusted p2p transactions with fuzzy reputation aggregation," *Internet Computing, IEEE*, vol. 9, pp. 24–34, Nov.-Dec. 2005.

[15] Z. Liang and W. Shi, "Pet: A personalized trust model with reputation and risk evaluation for p2p resource sharing," in *System Sciences. HICSS '05. Proceedings of the 38th Annual Hawaii International Conference on*, Jan. 2005.

[16] S. Lee, R. Sherwood, and B. Bhattacharjee, "Cooperative peer groups in nice," in *INFOCOM. Twenty-Second Annual Joint Conference of the IEEE Computer and Communications Societies.*, vol. 2, pp. 1272–1282, March-April 2003.

[17] A. A. Selcuk, E. Uzun, and M. R. Pariente, "A reputation-based trust management system for p2p networks," *International Journal of Network Security*, vol. 6, pp. 235–245, May 2008.

[18] H. Zhao and X. Li, "H-trust: A robust and lightweight group reputation system for peer-to-peer desktop grid," *Journal of Computer Science and Technology (JCST)*, vol. 24, pp. 833–843, Sep 2009.

[19] H. Zhao and X. Li, "Vectortrust: The trust vector aggregation scheme for trust management in peer-to-peer networks," in *The 18th International Conference on Computer Communications and Networks (ICCCN 2009)*, (San Francisco, CA USA), Aug 2-6 2009.

[20] H. Zhao and X. Li, "Vectortrust: Trust vector aggregation scheme for trust management in peer-to-peer networks," *The Journal of Supercomputing, Springer*, 2011.

[21] H. Zhao, X. Yang, and X. Li, "ctrust: Trust aggregation in cyclic mobile ad hoc networks," in *16th International European Conference on Parallel Processing (Euro-Par 2010)*, (Naples, Italy), Aug 31 - Sep 1 2010.

[22] K. Walsh and E. G. Sirer, "Experience with an object reputation system for peer-to-peer filesharing," in *NSDI'06: Proceedings of the 3rd conference on 3rd Symposium on Networked Systems Design*, (Berkeley, CA, USA), p. 1, 2006.

[23] M. Piatek, T. Isdal, A. Krishnamurthy, and T. Anderson, "One hop reputations for peer to peer file sharing workloads," in *Proceedings of the 5th USENIX Sym-*

posium on Networked Systems Design and Implementation, NSDI'08, pp. 1–14, 2008.

[24] M. Xie and H. Wang, "A collaboration-based autonomous reputation system for email services," in *The 29th IEEE Conference on Computer Communications (INFOCOM2010)*, (San Diego, CA), Mar 15-19 2010.

[25] S. Buchegger and J.-Y. L. Boudec, "A robust reputation system for mobile ad-hoc networks," technical report, IC/2003/50, EPFL-IC-LCA, 2003.

[26] S. Buchegger and J. Y. L. Boudee, "Self-policing mobile ad hoc networks by reputation systems," *Communications Magazine, IEEE*, vol. 43, no. 7, pp. 101–107, 2005.

[27] P. Michiardi and R. Molva, "Core: A collaborative reputation mechanism to enforce node cooperation in mobile ad hoc networks," in *Sixth IFIP conference on security communications, and multimedia (CMS 2002)*, (Portoroz, Slovenia), 2002.

[28] Y. L. Sun, W. Yu, Z. Han, and K. J. R. Liu, "Information theoretic framework of trust modeling and evaluation for ad hoc networks," *IEEE Journal on Selected Areas in Communications*, vol. 24, no. 2, 2006.

[29] S. Ganeriwal, L. K. Balzano, and M. B. Srivastava, "Reputation-based framework for high integrity sensor networks," *ACM Trans. Sen. Netw.*, vol. 4, no. 3, pp. 1–37, 2008.

[30] G. Theodorakopoulos and J. Baras, "On trust models and trust evaluation metrics for ad hoc networks," *IEEE Journal on Selected Areas in Communications*, vol. 24, no. 2, p. Feb, 2006.

[31] Q. Zhang and T. Yu, "On the modeling of honest players in reputation systems," in *Distributed Computing Systems Workshops. ICDCS '08. 28th International Conference on*, (Beijing, China), pp. 249–254, June 17-20 2008.

[32] Q. Feng, Y. Yang, Y. Sun, and Y. Dai, "Modeling attack behaviors in rating systems," in *Distributed Computing Systems Workshops. ICDCS '08. 28th International Conference on*, (Beijing, China), pp. 241–248, June 17-20 2008.

[33] K. A. Z. Despotovic, "P2p reputation management: Probabilistic estimation vs. social networks (management in peer-to-peer systems: Trust, reputation and security)," *Computer Networks*, vol. 50, no. 4, p. 485C500 (special issue), 2006.

[34] L. Becchetti, C. Castillo, D. Donato, S. Leonardi, and R. Baeza-Yates, "Using rank propagation and probabilistic counting for link-based spam detection," in *WebKDD* (A. Press, ed.), (Pennsylvania, USA), 2006.

[35] D. Donato, C. Castillo, M. Paniccia, G. Cortese, M. Selis, and S. Leonardi, "New metrics for reputation management in p2p networks," in *AIRWeb '07*, (Banff, Alberta, Canada), May 8 2007.

[36] Z. Liang and W. Shi, "Analysis of ratings on trust inference in open environments," *Elsevier Performance Evaluation*, vol. 65, pp. 99–128, Feb 2008.

[37] D. Yao, R. Tamassia, and S. Proctor, "Private distributed scalar product protocol with application to privacy-preserving computation of trust," in *Proceedings of IFIPTM-Joint iTrust and PST Conferences on Privacy, Trust Management and Security*, (Moncton, New Brunswick, Canada), Jul 2007.

[38] R. A. Shaikh, H. Jameel, B. J. dAuriol, H. Lee, S. Lee, and Y.-J. Song, "Group-based trust management scheme for clustered wireless sensor networks," *IEEE Transactions on Parallel and Distributed Systems*, vol. 20, Nov 2009.

[39] S. Marti and H. Garcia-Molina, "Taxonomy of trust: Categorizing p2p reputation systems," *Computer Networks*, vol. 50, pp. 472–484, Mar 2006.

[40] S. Ruohomaa, L. Kutvonen, and E. Koutrouli, "Reputation management survey," in *Second International Conference on Availability, Reliability and Security (ARES'07)*, 2007.

[41] Y. Sun, Z. Han, W. Yu, , and K. J. R. Liu, "A trust evaluation framework in distributed networks: Vulnerability analysis and defense against attacks," in *The 25th Conference on Computer Communications (IEEE INFOCOM'06)*, (Barcelona, Spain), April 23-29 2006.

[42] R. Jurca and B. Faltings, ""confess". an incentive compatible reputatiion mechanism for the online hotel booking industry," in *IEEE Conference on E-Commerce*, (San Diego, CA, USA), 2004.

[43] R. Jurca and B. Faltings, "Enforcing truthful strategies in incentive compatible reputation mechanisms," *Internet and Network Economics*, pp. 268–277, 2005.

[44] R. Jurca and B. Faltings, "An incentive-compatible reputation mechanism," in *IEEE Conference on E-Commerce*, (Newport Beach, USA), pp. 285–292, 03.

[45] M. Feldman, K. Lai, I. Stoica, and J. C. Robust, "Incentive techniques for peer-to-peer networks," in *4th ACM Conference on Electronic Commerce (EC04)*, (New York, NY, USA), May 2004.

[46] T. Papaioannou and G. Stamoulis, "An incentives' mechanism promoting truthful feedback in peer-to-peer systems," in *5th IEEE/ACM International Symposium in Cluster Computing and the Grid (CCGRID2005)*, (Cardi, UK), 2005.

[47] Z. Li and H. Shen, "Analysis of a hybrid reputation management system for mobile ad hoc networks," in *The 18th International Conference on Computer*

Communications and Networks (ICCCN 2009), (San Francisco, CA USA), Aug 2-6 2009.

[48] E. Fehr and S. Gachter, "Altruistic punishment in humans," *Nature*, pp. 137–140, Jan 2002.

[49] N. Miller, P. Resnick, and R. Zeckhauser, "Eliciting informative feedback: The peer-prediction method," *Management Science*, vol. 51, pp. 1359–1373, Sep 2005.

[50] E. Buchmann, K. B, and C. von der Weth, "Truthful answers are surprisingly common:experimental tests of the bayesian truth serum," *Lecture Notes in Computer Science*, vol. 4275, pp. 498–515, 11 2006.

[51] D. Prelec, "A bayesian truth serum for subjective data," *Science*, vol. 306, pp. 462–466, October 2004.

[52] Z. Zhang, S. Chen, and M. Yoon, "March: A distributed incentive scheme for peer-to-peer networks," in *The 26th IEEE Conference on Computer Communications (INFOCOM 2007)*, (Anchorage, Alaska, USA), May 2007.

[53] T. Moscibroda and S. Schmid, "On mechanism design without payments for throughput maximization," in *The 28th IEEE Conference on Computer Communications (INFOCOM 2009)*, (Rio de Janeiro, Brazil), April 2009.

[54] J. Park and M. van der Schaar, "Pricing and incentives in peer-to-peer networks," in *The 29th IEEE Conference on Computer Communications (INFOCOM 2010)*, (San Diego, CA, USA), March 2010.

[55] B. Chen and M. C. Chan, "Mobicent: a credit-based incentive system for disruption tolerant network," in *The 29th IEEE Conference on Computer Communications (INFOCOM 2010)*, (San Diego, CA, USA), March 2010.

[56] T. Chen and S. Zhong, "Inpac: An enforceable incentive scheme for wireless networks using network coding," in *The 29th IEEE Conference on Computer Communications (INFOCOM 2010)*, (San Diego, CA, USA), March 2010.

[57] H. Zhao, X. Yang, and X. Li, "Wim: A wage-based incentive mechanism for reinforcing truthful feedbacks in reputation systems," in *The 53rd Annual IEEE Global Telecommunications Conference (GLOBECOM 2010)*, (Miami, FL, USA), Dec 6-10 2010.

[58] C. Dellarocas, "Analyzing the economic efficiency of ebay-like online reputation reporting mechanisms," in *3nd ACM Conference on Electronic Commerce*, 2001.

[59] J. E. Hirsch, "An index to quantify an individual's scientific research output," 2005.

[60] wikipedia, "http://en.wikipedia.org/wiki/."

[61] D. Zhou and V. Lo, "Cluster computing on the fly: resource discovery in a cycle sharing peer-to-peer system," in *CCGRID '04: Proceedings of the 2004 IEEE International Symposium on Cluster Computing and the Grid*, pp. 66–73, 2004.

[62] G. Deen, T. J. Lehman, and J. H. Kaufman, "The almaden optimalgrid project," in *Active Middleware Services*, pp. 14–21, 2003.

[63] B. Michel, "General-purpose gpu computing: practice and experience," in *SC '06: Proceedings of the 2006 ACM/IEEE conference on Supercomputing*, p. 233, 2006.

[64] A. Luther, R. Buyya, R. Ranjan, and S. Venugopal, "Alchemi: A .NET-based grid computing framework and its integration into global grids," in *Technical Report, GRIDS-TR-2003-8, Grid Computing and Distributed Systems Laboratory, University of Melbourne, Australia, December 2003*, 2004.

[65] N. Noam and A. Ronen, "Algorithmic mechanism design," *Games and Economic Behavior*, vol. 35, pp. 166–196, April 2001.

[66] E. Rasmusen, *Games and information: An Introduction to Game Theory*. Blackwell Publishing, 2007.

[67] U. Wilensky, "Netlogo." Http://ccl.northwestern.edu/netlogo, 1999.

[68] V. Srinivasan, M. Motani, and W. T. Ooi, "Analysis and implications of student contact patterns derived from campus schedules," in *MobiCom '06: Proceedings of the 12th annual international conference on Mobile computing and networking*, (New York, NY, USA), pp. 86–97, 2006.

[69] V. Srinivasan, M. Motani, and W. T. Ooi, "CRAWDAD data set nus/contact (v. 2006-08-01)." Downloaded from http://crawdad.cs.dartmouth.edu/nus/contact, Aug. 2006.

[70] C. Liu and J. Wu, "Routing in a cyclic mobispace," in *MobiHoc: Proceedings of the 9th ACM international symposium on Mobile ad hoc networking and computing*, pp. 351–360, 2008.

VITA

Huanyu Zhao

Candidate for the Degree of

Doctor of Philosophy

Dissertation: TRUSTNET: A TRUST AND REPUTATION MANAGEMENT SYSTEM IN DISTRIBUTED ENVIRONMENTS

Major Field: Computer Science

Biographical:

Personal Data: Born in Benxi, Liaoning, China on Jan 08, 1983.

Education:

Received the B.E. degree from University of Science and Technology of China, Hefei, Anhui, China, 2006, in Information Security
Received the B.S. degree from University of Science and Technology of China, Hefei, Anhui, China, 2006, in Management Science
Completed the requirements for the degree of Doctor of Philosophy with a major in Computer Science Oklahoma State University in July, 2011.

Experience:

Mr. Huanyu Zhao is a Ph.D. candidate in the Department of Computer Science, Oklahoma State University (OSU). His research interests include Security & Privacy, Networking, and Cloud Computing. He has published 4 journal papers and more than 10 conference papers. He received the best paper award in the 2007 IEEE International Symposium on Ubisafe Computing (Ubisafe'07). Mr. Zhao was a student intern researcher in Mitsubishi Electric Research Laboratories (MERL), a graduate research assistant in Scalable Software Systems Laboratory (S3Lab), a teaching assistant in OSU, the president of ACM chapter at OSU. He received B.E. degree in Information Security and B.S. degree in Management Science (Dual Bachelors) from University of Science and Technology of China (USTC) in June 2006. Mr. Zhao is a recipient of Don and Shirley Fisher Graduate Scholarship.

He likes pingpang, photography, video game, and to play with his cat.

Name: Huanyu Zhao Date of Degree: July, 2011

Institution: Oklahoma State University Location: Stillwater, Oklahoma

Title of Study: TRUSTNET: A TRUST AND REPUTATION MANAGEMENT SYSTEM IN DISTRIBUTED ENVIRONMENTS

Pages in Study: 94 Candidate for the Degree of Doctor of Philosophy

Major Field: Computer Science

With emerging Internet-scale open content and resource sharing, social networks, and complex cyber-physical systems, trust issues become prominent. Despite their rigorous foundations, conventional network security theories and mechanisms are inadequate at addressing such loosely-defined security issues in decentralized open environments. In this dissertation, we propose a trust and reputation management system architecture and protocols (TrustNet), aimed to define and promote trust as a first-class system parameter on par with communication, computation, and storage performance metrics. To achieve such a breakthrough, we need a fundamentally new design paradigm to seamlessly integrate trust into system design. Our TrustNet initiative represents a bold effort to approach this ultimate goal.

TrustNet is built on the top of underlying P2P and mobile ad hoc network layer and provides trust services to higher level applications and middleware. Following the TrustNet architecture, we design, implement, and analyze trust rating, trust aggregation, and trust management strategies. Especially, we propose three trust dissemination protocols and algorithms to meet the urgent needs and explicitly define and formulate end-to-end trust. We formulate trust management problems and propose the H-Trust, VectorTrust, and cTrust scheme to handle trust establishment and aggregation issues. We model trust relations as a trust graph in distributed environment to enhance accuracy and efficiency of trust establishment among peers. Leveraging the distributed Bellman-Ford algorithm, stochastic Markov chain process and H-Index algorithm for fast and lightweight aggregation of trust scores, our scheme are decentralized and self-configurable trust aggregation schemes.

To evaluate TrustNet management strategies, we simulated our proposed protocols in both unstructured P2P network and mobile ad hoc network to analyze and simulate trust relationships. We use software generated data as well as real world datasets. Particularly, the student contact patterns on the NUS campus is used as our trust communication model. The simulation results demonstrate the features of trust relationship dissemination in real environments and the efficiency, accuracy, scalability and robustness of the TrustNet system.

ADVISOR'S APPROVAL: _____