Trust beyond reputation : novel trust mechanisms for distributed environments

Liu, Xin

2011

Liu, X. (2011). Trust beyond reputation : novel trust mechanisms for distributed environments. Doctoral thesis, Nanyang Technological University, Singapore.

<https://hdl.handle.net/10356/46445>

<https://doi.org/10.32657/10356/46445>

Trust beyond reputation: Novel trust mechanisms for distributed environments

Liu Xin

School of Computer Engineering

A thesis submitted to the Nanyang Technological University
in fulfillment of the requirement for the degree of
Doctor of Philosophy

2011

Abstract

In recent years, people have taken a more active role in participating in massively distributed systems (e.g., Web 2.0 applications) by intensively interacting with other system participants. Such rich interactions enhance Internet experience significantly. However, due to the characteristics of such systems like openness, scale, heterogeneity, etc., it is difficult to ensure smooth/reliable interactions among the participants. Trust management, which aids automated decision-making process or alternatively provides decision support, is a popular approach to help participants to select the trustworthy interaction partners. Traditional reputation based trust mechanisms may be effective in certain scenarios, but they suffer from two main issues: (i) they often rely upon knowledge (i.e., historical information about past behavior of a specific agent) that may not actually be available or accurate to the assessor. (ii) Even if the required historical information is available, current mainstream methods simply investigate outcomes of the target agent's past transactions to derive a single trust/reputation value (and sometimes, an additional confidence metric on that value), thus failing to detect dynamic behavior of the intelligent agents who may strategically change his behavior to maximize profits.

It is thus essential to develop novel mechanisms that support “trust beyond reputation”. For the issue (i), we first introduce a stereotyping based computational trust model StereoTrust. The trustor (i.e., the agent that needs to derive trust on other agents) forms stereotypes using his previous transactions with other relevant agents. When facing a stranger, the stereotypes matching stranger's profile are aggregated to derive the expected trust. StereoTrust uses an intuitive and heuristic method to derive and combine stereotypes for trust assessment. In order to improve StereoTrust

by discriminating different types of stereotypes, we then develop MetaTrust, a machine learning techniques based trust framework. The agent uses his own previous transactions (with other agents) to build a knowledge base, and utilize this to assess the trustworthiness of a transaction based on the associated features that are harnessed using machine learning techniques. In addition, we propose to construct a local knowledge sharing overlay network (LKSON) to bootstrap the inexperienced agents who have limited local knowledge.

In order to validate usefulness of the stereotype based trust mechanisms in reality, we apply them to two different massively distributed applications: (1) Peer-to-Peer backup storage systems (i.e., a decentralized setting) for selecting suitable data holders and (2) Internet online auction site (i.e., a centralized setting) for detecting auction frauds. Simulation results show that in both settings, our approaches work effectively and efficiently in comparison to other existing approaches.

For the issue (ii), we develop solutions to capture dynamic behavior of the target agent based on his past transactions in two aspects: (1) we propose a context matching based approach to detect the target agent's past behavior patterns for trust assessment; (2) we propose the concept of *imprudence* to study and detect the inappropriate behavior of an agent with high reputation value and relatively long history, and hence considered 'reliable' by traditional reputation systems.

Keywords: distributed environments, trust, stereotypes, features, machine learning, imprudence

Acknowledgments

There are a lot of people I would like to thank for a variety of reasons. First of all, I would like to thank my supervisor Dr. Anwitaman Datta, for giving me the opportunity to pursue Ph.D. at NTU, for his unending patience and guidance. He is not only a wonderful supervisor but also a caring friend. This thesis could not have been completed without his full support.

I am grateful to my other co-authors: Dr. Krzysztof Rzdca, Dr. Gilles Tredan, Dr. Adam Wierzbicki, Dr. Lim Ee Peng, Mr. Tomasz Kaszuba, Mr. Radoslaw Nielek and Dr. Wu Di. The opportunity to work with them has been rewarding and learning experiences. Particular thanks is given to Dr. Adam Wierzbicki for hosting me during my visit to PJWSTK, and Mr. Tomasz Kaszuba and Mr. Radoslaw Nielek for their warm reception.

I would like to thank Dr. Zhang Jie and Dr. Karl Aberer for their valuable suggestions for my work. In particular, Dr. Zhang Jie's deep understanding of trust provides me helpful guidance for my future research.

I would also thank Dr. Sourav Bhowmick, director of Centre for Advanced Information Systems (CAIS), for ensuring a conducive research environment in the centre, and the support staffs (Lai Chee Keong, Chua Chiew Song, Ang Ah Giat and Loo Kian Hock), for the technical and administrative support.

Finally, I must thank my friends and family. In particular, I am grateful to my parents for their love and encouragement. And most of all, to my wife Ma Ting whose love has kept me going, even when I doubted myself. Without them, I have no way to be who I am today.

Contents

Abstract	i
Acknowledgments	iii
List of Figures	ix
List of Tables	xii
 I Preliminaries	 1
1 Introduction	2
1.1 Overview	2
1.2 Problem statement	4
1.3 Research scope and solutions	6
1.3.1 Stereotyping based trust models	6
1.3.2 Applications of the stereotyping based trust models	9
1.3.3 Trust mechanisms capturing agents' dynamic behavior	10
1.4 Research contributions	11
1.5 Dissertation outline	14
 2 Background and related works	 16
2.1 Trust management	16
2.1.1 Definition of trust	16
2.1.2 Definition of reputation	18
2.2 Trust and reputation systems for distributed environments	19
2.2.1 A Distributed Trust Model	19

2.2.2	EigenTrust	20
2.2.3	PeerTrust	22
2.2.4	P2PRep	23
2.2.5	PowerTrust	24
2.2.6	Beta reputation system and TRAVOS	26
2.2.7	Stereotyping based approaches	29
II	Trust based on stereotyping	31
3	StereoTrust: a stereotypes based personalized trust model	32
3.1	Introduction	33
3.2	Description of stereotyping based Trust Model	36
3.2.1	Group Definition	36
3.2.2	Modeling Trust	37
3.2.3	StereoTrust Model	38
3.2.4	Dichotomy Based Enhanced Model	40
3.3	Evaluation	44
3.3.1	Methodology	44
3.3.2	Epinions Dataset	47
3.3.3	Synthetic Dataset	51
3.4	Conclusions	54
4	A machine learning framework for identifying relevant features to determine trust	57
4.1	Introduction	58
4.2	System model	60
4.2.1	Notation	60
4.2.2	Feature collection	60
4.2.3	Architecture	62

4.3	Machine learning algorithms for trust assessment	64
4.3.1	Case study: Using linear discriminant analysis	64
4.3.2	Case study: Using decision tree	67
4.4	Algorithm recommendation confidence	70
4.4.1	Measuring recommendation confidence for LDA based algorithm .	70
4.4.2	Measuring recommendation confidence for DT based algorithm .	71
4.5	Local Knowledge Sharing Mechanism	73
4.5.1	Local Knowledge Sharing Overlay Network (LKSON)	73
4.5.2	LKSON Update	78
4.6	Conclusion	79

III Applications of stereotyping based trust 81

5 Enhancing Data Availability for Peer-to-Peer Backup 83

5.1	Introduction	84
5.1.1	Motivation	85
5.1.2	The proposed scheme	86
5.1.3	Scope	87
5.1.4	Summary	88
5.2	Background	88
5.2.1	System model	88
5.2.2	Modeling trust	89
5.3	Contextual trust based data placement scheme	90
5.3.1	Preliminaries	90
5.3.2	Local knowledge management	91
5.3.3	Collecting contextual meta-information about potential data holders	92
5.3.4	Trust calculation	94
5.3.5	Algorithm outline: Selection of data holders	94
5.4	Evaluation	95

5.4.1	Simulation environment and methodology	95
5.4.2	Results	101
5.5	Related work	109
5.6	Conclusion	113
6	Identifying Risky Transactions in Internet Auctions	115
6.1	Introduction	116
6.2	MetaTrust based approach	118
6.3	StereoTrust based approach	119
6.3.1	Identifying Attributes	119
6.3.2	Creating Stereotypes Using Identified Attributes	120
6.3.3	Local Stereotypes versus Community Stereotypes	122
6.3.4	Estimating Trust of Transaction Using Stereotypes	123
6.4	Evaluation	125
6.4.1	Methodology	126
6.4.2	Results	128
6.5	Conclusion	137
IV	Trust capturing dynamic behavior	139
7	Detecting agents' dynamic behavior based on context matching	141
7.1	Introduction	142
7.2	Our approach	143
7.2.1	Notation	143
7.2.2	Using direct experience	144
7.2.3	Incorporating indirect experience	148
7.2.4	Computational complexity	148
7.3	Evaluation	149
7.3.1	Simulation settings	149

7.3.2	Results	151
7.4	Conclusion	154
8	Detecting Imprudence of ‘Reliable’ Sellers in Online Auction Sites	156
8.1	Introduction	157
8.1.1	Motivation	157
8.1.2	Problem Statement	159
8.1.3	Approach	160
8.2	Methodology	161
8.2.1	Notations	161
8.2.2	Features	161
8.3	Prediction strategy	166
8.3.1	Training Data	166
8.3.2	Learning Algorithm	168
8.3.3	Prediction Algorithm	168
8.4	Evaluation	170
8.4.1	Experiment Methodology	171
8.4.2	Results	172
8.5	Conclusion	177
V	Concluding remarks and future directions	179
9	Conclusion	180
9.1	Summary	180
9.2	Future works	183
9.2.1	Extensions of the stereotyping based trust mechanisms	184
9.2.2	More application areas	185
9.2.3	Improving the proposed trust mechanisms for capturing an agent’s dynamic behavior	186
9.2.4	Implementation and deployment	187

List of Figures

2.1	Examples of beta PDF.	27
3.1	Trust computation flow: reputation based trust mechanisms versus StereoTrust	33
3.2	Process of trust calculation. Weighted sum of each group G^i 's trust by assigning corresponding weight factor $W_{x,y}^i$	40
3.3	Structure of group G^i in d-StereoTrust	41
3.4	Process of trust calculation. Trusts of honest subgroup $G^{i,h}$ and dishonest subgroup $G^{i,d}$ of each group G^i are firstly combined using closeness and then trusts of all groups G^i are combined using weight factor $W_{x,y}^i$ to derive the target agent's trust.	42
3.5	Experiments results using Epinions.com dataset.	50
3.6	Experiments results using synthetic dataset.	53
4.1	The architecture.	62
4.2	A local knowledge sharing overlay network.	74
5.1	Node availability	96
5.2	Real PlanetLab trace ($M = 8, N = 32$)	102
5.3	Messages sent for retrieving required files (real PlanetLab trace: $M = 8,$ $N = 32$).	104
5.4	Real PlanetLab trace ($M = 16, N = 32$)	106
5.5	Synthetic trace ($M = 8, N = 32$)	107
5.6	Synthetic trace ($M = 16, N = 32$)	108

5.7	Important design decisions that need to be considered when designing and deploying a P2P storage system	109
5.8	Wuala architecture	110
6.1	Sub groups of group G_{c_i} and distances between potential transaction and the two sub groups.	121
6.2	Deriving trust of potential transaction by combining corresponding stereotypes.	125
6.3	False positive rate versus false negative rate (StereoTrust based approach).129	
6.4	Overall falseness (StereoTrust based approach).	130
6.5	Comparison of performance of local stereotypes and community stereotypes (StereoTrust based approach).	131
6.6	Experiments using Allegro dataset (MetaTrust based approach).	133
6.7	Performance of MetaTrust based approach (LDA) with LKSON.	134
6.8	Performance comparison using synthetic dataset ($\theta' = 0.2$).	135
6.9	Performance comparison using synthetic dataset ($\theta' = 0.4$).	135
6.10	Performance comparison using synthetic dataset ($\theta' = 0.6$).	135
6.11	Performance comparison using synthetic dataset ($\theta' = 0.8$).	136
7.1	Measuring similarity between recent transaction window and earlier transaction window.	147
7.2	Experiments using Allegro dataset (one transaction window size).	151
7.3	Experiments using Allegro dataset (multiple transaction window sizes).	152
7.4	Our approach vs PTE (synthetic dataset, multiple transaction window size).	152
8.1	Cumulative distribution function (CDF) of the reputation value for unsuccessful transactions	158
8.2	Imprudence detection workflow	161
8.3	Features verification	163

8.4	Evaluation of individual sellers	169
8.5	Accuracy versus coverage (Allegro dataset)	175

List of Tables

3.1	Mean Absolute Error (with 95% Confidence Interval)	50
3.2	Mean Absolute Error and Coverage (for synthetic dataset)	53
4.1	Structure of local knowledge repository	63
5.1	Notations summary	90
5.2	Structure of Transaction Database of Data Owner p_o	92
5.3	Structure of Meta-information Database of Data Owner p_o	92
5.4	Simulation Parameters	99
6.1	Choosing Attributes	120
6.2	Summary of overall falseness rates and the corresponding thresholds (StereoTrust based approach)	131
8.1	Summary of notations	162
8.2	Feature verification. ‘R’ stands for reputation, ‘A’ stands for previous transactions volume, ‘T’ stands for time of the transaction happened and ‘P’ stands for item price.	173
8.3	Performance comparison	176

Part I

Preliminaries

Chapter 1

Introduction

1.1 Overview

Many internet-scale applications have evolved recently, fueled by and fueling applications open to all, e.g., Web 2.0 and Peer-to-Peer (P2P) applications, where end-users play active role in the system, and thus their activities in turn significantly impact the system's behavior and other users' experience with the same. For instance, different from traditional static web sites, Web 2.0 applications allow users to not only retrieve information but also contribute contents and share information with others (e.g., Facebook [1], eBay [2], Flickr [3], Twitter [4], Wikipedia [5]). On the other hand, Peer-to-Peer (P2P) networks [6] enable applications like content distribution (e.g., Kazaa[7], BitTorrent [8], eMule [9]), distributed computation (e.g., Seti@home [10], Genome@home [11]), etc. by establishing direct connections between users. Such rich interactions among users enhance the Internet experience significantly. However, due to the characteristics of massively distributed systems such as openness, scale, heterogeneity, etc., dishonest users can easily join the system and act maliciously or selfishly to achieve their goals. This thus arises serious security issue: how to ensure that the (unknown) potential interaction partner will not harm the interests of the honest user in the transactions.

One solution for promoting successful transactions between a pair of users is to rely on dedicated security infrastructure such as authentication, authorization and accounting (AAA) [12] and Public Key Infrastructure (PKI) [13]. AAA is an architectural framework for configuring a set of three independent security functions in a consistent

manner. Authentication provides a way of identifying a user, typically by having the user enter a valid token (e.g., username/password) before access to the system is granted. Authorization is the process of granting or denying a user access to network resources once the user has been authenticated. Accounting is the process of keeping track of a user's activity while accessing the network resources. On the other hand, PKI is the implementation of various security mechanisms (e.g., confidentiality, integrity, authentication and non-repudiation), which binds public keys with respective user identities by means of a certificate authority (CA). Both AAA and PKI indeed help in reducing the impact of behavior of the dishonest users. However, most current designs of these two security mechanisms have used an administrator-centric model (i.e., relying on centralized root authority), which can become a single point of failure or attack (physical and/or legal), a bottleneck, incur high administrative overhead or be a point of censorship or monopoly or may not scale. More importantly, AAA and PKI belong to "hard security", which only helps establish that the interaction partner is authenticated and authorized to take various actions but do not ensure that that party is doing what is expected and delivering good service, i.e., they only assume complete certainty. So an alternative "soft" approach is needed to deal with uncertainty (e.g., we never know other users' intentions; we cannot tell the user we are going to interact with has sufficient competence and resources, etc.) that is quite common in massively distributed systems. *Trust*, in this case, is a kind of "soft security" [14] we need.

The concept of trust is prevalent in our society [15, 16]. From medical treatment to business trade, we must establish certain level of trust before acting. Increasingly, with the development of Internet and World Wide Web (WWW), more and more people interact with others in the virtual world, the need of trust is thus widespread in various areas of computer science. In the domain of massively distributed systems, trust management plays an important role in decision making. For instance, in online auction sites, buyer must ensure that the seller will not cheat in the transaction like selling low quality item, the item is not properly packaged, delivery of item is delayed, etc.

Numerous works are proposed to manage trust relations among agents in distributed environments [17, 18, 19, 20, 21, 22, 23, 24, 25, 26, 27, 28]. One popular method is to rely on past behavior of the specific agent in question. The basic idea of such a method is to let agents involved in the interactions assess (e.g., rate) each other. For instance, after an interaction, the service requestor rates the service provider according to his experience in the interaction. Such ratings are used by this service requestor as well as other agents to determine whether or not to interact with that service provider in the future encounters. Generally, if an agent mostly performs honestly in the past, he is considered trustworthy in the future interactions, otherwise, he is considered untrustworthy.

1.2 Problem statement

Traditional reputation based trust management mechanisms¹ derive trust relying on past behavior of the specific agent in question. Normally, there are two ways for a trustor to learn the target agent's past behavior, one is *direct experience*, i.e., the direct interactions between the trustor and the target agent. If a trustor has sufficient direct experience with the target agent, his future performance can be reasonably predicted [31]. However, in large-scale environments, direct experience is often not sufficient or even non-existent. In this case, the other way to obtain the target agent's past behavior is to rely on *indirect experience*, which is the opinions obtained from other agents who have already interacted with the target agent. The third parties' opinions can be simply aggregated just like a seller's ratings on eBay [32], or combined using more sophisticated weight mechanisms [23, 33, 34, 20, 35]. Alternatively, transitive trust models [17, 36, 37, 38] (or web of trust models) build chains of trust relationships between the trustor and the target agent. The basic idea is that if A knows B and B knows C , then A can derive C 's trust using B 's referral on C and A 's trust in B .

Current reputation based trust mechanisms suffer from two main drawbacks:

---

¹It is worth mentioning that besides reputation, there are several other ways to derive trust such as role based method [29, 30]. However, this thesis only focuses on the popular reputation based approach, i.e., assessing trust relying on an agent's past behavior (e.g., interactions with other agents).

- (1) As mentioned above, direct experience is not often available to the assessors. Furthermore, collecting correct indirect experience efficiently in a large scale, hostile environment is non-trivial. Agents may refuse to give their opinions on target agents due to privacy issues or provide inaccurate opinions. Lots of works are proposed to handle such unfair opinions, but most of them have their limitations: some of them do not scale well [17, 39], some of them rely heavily on certain network topology [25, 28] and some of them incur high overhead in terms of computation, storage and communication [40, 26]. Moreover, we note that even if an agent’s historical information is available through indirect sources, unlike traditional scenarios, “good” and “bad” may have different interpretations in a massively distributed environment. Agents may inherently be dishonest, e.g., cheating interaction partner in various ways, and such behavior can be modeled using traditional trust models. However, agents may also be inherently well-behaved, but may be suitable for some agents and yet be unsuitable for some others. For instance, in order to find out how is the Chinese food served by a restaurant, a Chinese customer a requests reviews from other customers who had dinner here (e.g., via a restaurant review site). Many customers like the Chinese food in this restaurant but unfortunately do not realize the food here is not authentic (e.g., these customers come from America, and the food has been localized). So if the reviews of the restaurant are aggregated from these customers to help a estimate quality of the Chinese food, the restaurant will appear trustworthy even though it is not a good place for enjoying real Chinese food (although the reviews are genuine), and there is nothing inherently wrong with the restaurant either. However, if inputs from ‘likeminded’ reviewers could be isolated, a more personalized recommendation could be derived.

- (2) Even if the complete historical information of the target agent is available, the current mainstream approach just statistically investigates outcomes of the target agent’s past transactions. Although numerous improved works have been

proposed, most of them follow the basic rule that the more the past successful transactions, the higher the probability that the target agent will act honestly in the near future, subject to any context constraints. Such a rule, of course, is true in certain scenarios, e.g., on eBay, a seller with over 99.9% positive feedbacks is indeed more reliable than those who are frequently given negative feedbacks by the buyers. However, this rule assumes a relatively static behavior thus is not able to capture an agent's dynamic behavior, which is common in a large scale, open system since self-interested agents are very likely to vary behavior to achieve their goals. For instance, a malicious seller may act honestly in selling cheap items to gather sufficient reputation and then cheat in selling an expensive item. Traditional reputation based trust mechanisms are likely to incorrectly predict the risky transaction as safe since a majority of the seller's past transactions were good.

It is thus essential to develop novel trust mechanisms that support “trust beyond reputation”.

1.3 Research scope and solutions

The research works presented in this thesis for addressing the issues discussed above are divided into three parts: (1) stereotyping based generic trust mechanisms developed for massively distributed systems without target agents' historical information, (2) application of the proposed trust mechanisms in two specific distributed settings: (i) P2P storage systems and (ii) Internet auction sites and (3) trust mechanisms aiming at capturing an agent's dynamic behavior based on his past transactions.

1.3.1 Stereotyping based trust models

As described in the previous section, trust management is indispensable in massively distributed systems to promote smooth transactions but current trust based approaches have their obvious drawbacks which make them unusable in some scenarios, e.g., the historical information that is used to derive trust is actually not available. This forms

the motivation of this thesis. In particular, we aim to *develop novel computational trust models to help estimate trustworthiness of the potential interaction partners without their historical information*. Our objective is to help trustor to select the “good” interaction partners who are not only honest (i.e., do not act maliciously or selfishly) but also suitable (i.e., provide trustor the relatively good service that may not be good to other agents). To achieve this goal, departing from traditional trust mechanisms, our approach should handle other kinds of knowledge locally available to a trustor, instead of the historical information of the target agent. Therefore, two main (progressive) issues must be addressed:

- (1) given an experienced agent (i.e., he has past experience with other agents), how does he organize his local knowledge to derive trustworthiness of an unknown interaction partner.
- (2) given an inexperienced agent (i.e., he has limited local knowledge), how does he collect suitable knowledge to derive trustworthiness of an unknown interaction partner.

To address the first issue, we use a different kind of trust information, i.e., stereotypes. A stereotype contains certain features of the agents/transactions and an expected outcome of the transaction. A trustor forms stereotypes by aggregating information from his interaction partners’ profiles, or the context of the transaction. To build stereotypes, the trustor has to group the agents he has already interacted with according to the features. Then, when facing an unknown agent, the trustor estimates his trust using stereotypes on the groups to which the unknown target agent belongs.

In this thesis, two techniques are used to organize a trustor’s local knowledge: Bayesian approach [41] and machine learning techniques [42]. Bayesian approach calculates stereotypes by taking binary ratings as input (i.e., we assume the outcome of a transaction is successful or unsuccessful). This approach is based on computing trust

scores by statistically updating beta probability density function (PDF). The posterior stereotype score is computed by combining a priori stereotype score with the new evidence [43, 19, 23]. The proposed trust model StereoTrust [44] is based on Bayesian approach. Alternatively, in order to explicitly differentiate the discriminating stereotypes from the useless ones, we propose a generic trust framework MetaTrust [45, 46] that is based on machine learning techniques. A trustor uses his own previous transactions (with other agents) to build a knowledge base, and utilizes this to assess the trustworthiness of a potential transaction based on the associated features, which are capable of distinguishing successful transactions from unsuccessful ones. These features are harnessed using appropriate machine learning algorithms (e.g., linear discriminant analysis [47] and decision tree [48]) to extract relationships between the potential transaction and previous successful/unsuccessful transactions. For both StereoTrust and MetaTrust, trust assessment is done according to a trustor’s local personal stereotypes, so the system is completely distributed and the obtained trust predictions are personalized.

Both StereoTrust and MetaTrust work under the assumption that a trustor has sufficient local knowledge (i.e., past experience with other agents). However, in many scenarios (e.g., online auction sites), inexperienced agents who only have limited local knowledge dominate the system, which restricts the application of our proposed trust mechanisms. This arises the second issue we mentioned before. One possible solution to this issue is to explore the social relations among the agents in the network to collect useful information. For instance, the inexperienced agents may only request knowledge from their “familiar” agents (e.g., friends, colleagues, etc). However, in some cases, there may not exist any such “familiar” agents or these agents may have different stereotypes thus providing inaccurate information. To address this problem, we construct local knowledge sharing overlay network (LKSON) on top of the current network infrastructure (e.g., P2P network) to share the local knowledge. To ensure that local knowledge is shared efficiently, some problems like how to filter out useless information, how to

protect privacy, what kind of local knowledge is shared and how to minimize network resources consumption, etc. are addressed.

1.3.2 Applications of the stereotyping based trust models

To validate the proposed trust mechanisms, we apply them to two different distributed settings to solve problems in specific scenarios. One is a decentralized setting, i.e., P2P storage systems. P2P storage services are cost-effective alternative for data backup [49, 50, 51, 52, 53]. A basic question that arises in such systems' design is: *In which peers do we store redundant data?* Choosing appropriate peers for data backup is important at a microscopic level, from an end-user's perspective to guarantee good performance, e.g., quick access, high availability, etc., as well as at a macroscopic level, e.g., for system optimization, fairness, etc. Existing systems apply different techniques, including random selection [54, 55, 56], as well as more sophisticated placement schemes and heuristics, e.g., based on a distributed hash table (DHT) [57, 58, 59, 60] or based on peers' trustworthiness, typically derived using peers' historical behavior [61, 62], and more recently, also by translating real-life social trust into storage system trust [63]. Based on StereoTrust model, we propose as an alternative, a contextual trust based data placement scheme to select suitable data holders. It is originally designed for and applicable to scenarios where there is inadequate historical information about peers, a common scenario in large-scale systems. Specifically, our scheme estimates trustworthiness of a peer based on stereotypes, formed by aggregating information of interactions with other (similar) peers. That is, when an unknown peer is encountered as a potential data holder, our scheme enables a data owner (or the system) to estimate the encountered peer's trust by categorizing it into different groups (stereotyping) according to relatively easier to obtain non-historical meta-information, e.g., time zone, location, Internet service provider, etc.

The other is a centralized setting, i.e., Internet auction sites. Encountering unknown sellers is very common in online auction sites. In such a scenario, a buyer could not

estimate the trustworthiness of the unknown seller based on the seller’s past behavior (like current reputation system [2]). The buyer is thus exposed to the risks of being cheated. We apply the proposed trust mechanisms to determine the risk of a potential transaction even if the seller is personally unknown to not only the buyer but also to the rest of the system. Specifically, our approach first identifies discriminating attributes which are capable of distinguishing successful transactions from unsuccessful ones. A buyer can then use his own past transactions (with other sellers) to form stereotypes according to the identified attributes.

1.3.3 Trust mechanisms capturing agents’ dynamic behavior

StereoTrust and MetaTrust partly solve the cold-start problem (i.e., historical information of the target agent is not available) of the trust/reputation systems. However, in a (more) common scenario, such historical information indeed exists. We thus investigate how to efficiently organize the target agent’s past transactions to capture his dynamic behavior, which cannot be effectively handled by traditional reputation based trust mechanisms. We address this problem in two aspects: (1) we study the target agent’s long term past behavior to try to detect patterns to guide current trust prediction; (2) we particularly focus on the target agent’s behavior deviation when he is considered ‘reliable’ by the existing reputation systems.

For the first aspect, we propose a context matching based trust prediction approach. The main idea is that outcome of the potential transaction can be indicated by that of certain previous transactions, which have the most similar transaction context. Specifically, given a set of a trustor’s past transactions (by chronological order) with the target agent, we first define context of a transaction as a sequence of its immediate previous transactions (a transaction window), each of which is described by a feature vector. We then compare similarity of the context of the potential transaction and that of the previous transactions. The outcome of a specific previous transaction that has the most similar context with that of the potential transaction is used as the indicator to estimate trustworthiness of the potential transaction.

For the second aspect, we propose the concept of *imprudence* to study and detect the inappropriate behavior of an agent with high reputation value and relatively long history, and hence considered ‘reliable’ according to conventional reputation systems. Specifically, we first identify and verify the factors that influence an agent’s imprudence behavior. Such features are harnessed using data mining algorithms to extract the relationships between contexts of the imprudence of other agents and that of the target agent. We then combine these factors using logistic regression to predict and study the probability of imprudence of a target agent.

1.4 Research contributions

The rationale for our research is to determine alternative and complementary trust mechanisms (than existing reputation based techniques) to compute trust regardless of the availability of the target agent’s historical information. The main contributions of this thesis are summarized as follows:

- (1) We propose StereoTrust [44], a computational trust model inspired by real life stereotypes. An agent forms stereotypes using his previous transactions with other agents. A stereotype contains certain features of the agents and an expected outcome of the potential transaction. These features can be taken from an agent’s profile information. When facing a stranger, the stereotypes matching stranger’s profile are aggregated to derive his expected trust. The stereotypes are based entirely on the local perspective and local information of the trustor, and, therefore, are naturally suited for large-scale systems; personalized for each trustor; and less susceptible to false or unsuitable information from third parties.
- (2) When third parties’ opinions about the target agent are available, we propose an enhanced version of StereoTrust (called d-StereoTrust), which creates a “good” and a “bad” subgroup inside each stereotype. The trustor assigns each of his previous transaction partners to one of these groups based on his personal experience

with the partner (e.g., the ratio of failed transactions). Then, the trustor uses the aggregated third parties' opinions about the unknown target agent to determine how similar is the agent to the "good" and "bad" subgroup. Third parties' opinions are a small subset of information used by traditional mechanisms (such as feedback aggregation [19, 27] or Eigentrust-type algorithms [25]).

- (3) StereoTrust relies on Bayesian approach to derive/update stereotypes and combine the same using an intuitive weight mechanism. Such a method is intuitive, heuristic and fails to tell which stereotypes are more important than others. To improve StereoTrust by organizing local knowledge more efficiently, we propose MetaTrust [45, 46], a generic trust framework for large-scale open systems using machine learning techniques. Our approach is generically based on machine learning algorithms, which provide classification of a potential transaction by learning a set of past transactions. In this work we instantiate the framework using two common machine learning algorithms, i.e., linear discriminant analysis and decision tree.
- (4) In order to address the issue of unavailability of local knowledge, in which case the proposed trust mechanisms cannot be applied, we develop a local knowledge sharing overlay network (LKSON) on top of the current network infrastructure (e.g. P2P). We propose a set of mechanisms to share local knowledge efficiently. We discuss how to combine the third party knowledge to derive trust using MetaTrust as an illustrative example.
- (5) We look at the issue of selecting suitable data holders without their complete historical information in P2P storage systems. Our scheme uses stereotypes to estimate contextual trust of the potential data holders. To build stereotypes, data owner groups other peers it has already interacted with according to some easier to obtain contextual meta-information of the peers. We identify two kinds of meta-information, which greatly influence selection of the suitable data holders:
  - (1) Internet Service Provider (ISP). Peers controlled by certain ISP provide fast

data transfer rate while peers controlled by other ISP do not. (2) Time zone. The peers that are in the same/similar time zones demonstrate the similar availability pattern thus significantly reducing the relative churn. So relying on our stereotyping based approach, data owners are able to select suitable data holders (to store data redundancy) who demonstrate relatively high availability and provide fast data access thus ensuring high data availability.

- (6) We apply the proposed trust mechanisms to detect auction fraud in Internet auction sites. That is, we consider the problem of predicting trustworthiness of a potential transaction where the seller is unknown to the buyer. Different from traditional historical information based approach, our approach uses a different kind of information: that of semantic similarity of the potential transaction to other past transactions that the buyer knows. When facing a potential transaction with an unknown seller, the buyer builds its trust by cumulating the experience from the stereotypes to which the potential transaction conforms. By studying Allegro dataset [64], we identify three useful classes of attributes to form stereotypes: (1) category of the item sold in the transaction, (2) price of the item and (3) number of items already sold by the seller when the transaction is conducted. We also investigate the accuracy of local knowledge and community knowledge.
- (7) We propose a trust prediction approach based on the target agent's historical information, aiming at capturing his dynamic behavior pattern to accurately estimate trustworthiness of the potential transaction with this agent. This work exploits the transaction context which is described by a set of features to find the most similar previous transaction whose outcome is used to predict outcome of the potential transaction.
- (8) From a real online auction dataset, we determined that imprudent behavior by the agents with high reputation is indeed a problem that exists in practice. We then developed an approach that looks beyond the average behavior, focusing on an

agent’s behavior at the specific stages of his life cycle, aided by carefully identified features.

Note that in order to validate effectiveness of our approach, we conduct experiments using both real datasets and synthetic datasets. This is because real data provides a realistic simulation environment, the obtained results are also relatively realistic and convinced. However, given various limitations of existing real datasets² (e.g., in Chapter 3 and 6, Epinions dataset and Allegro dataset provide excessively friendly environments; in Chapter 5, PlanetLab trace only contains limited amount of nodes; in Chapter 7, Allegro dataset lacks rich behavior patterns of the agents), our approaches are only partially evaluated in the specific scenarios. So in order to comprehensively evaluate our approaches, we generate synthetic datasets on the basis of the corresponding real datasets to provide more generic domains while preserving reality of the simulation environments.

1.5 Dissertation outline

In the first part of this thesis, we provide the preliminary knowledge. After this chapter which presents the overview, motivating context and brief summary of research, we introduce background information and review related works in Chapter 2. We first describe the notations of trust and reputation, and then review the existing trust management mechanisms developed for distributed systems.

In the second part, we present StereoTrust model and MetaTrust framework. In Chapter 3, basic StereoTrust, which only relies on trustor’s local knowledge is presented. We also propose an enhancement of the basic model called d-StereoTrust when third parties’ opinions about the target agent are available. After that, we present MetaTrust in Chapter 4, which more reasonably organizes the trustor’s local knowledge to predict the trustworthiness of a potential transaction. We also introduce the local knowledge

---

²The lack of suitable real datasets for evaluation is partially due to the fact that our approaches use different kinds of information and identify new behavior patterns, which have not been thoroughly studied in literature.

sharing overlay network (LKSON) to efficiently propagate local knowledge. We discuss the issues of what kind of local knowledge is shared, how to protect privacy, how to filter out useless information, how to combine the third parties' knowledge to derive trust and so on.

In the third part of the thesis, we describe how the proposed trust mechanisms are applied in concrete application scenarios to solve specific problems. In P2P storage systems (Chapter 5), we introduce contextual trust based data placement scheme to help data owners select suitable data holders. In Internet auction sites (Chapter 6), we develop novel approach to detect auction fraud in absence of target seller's past behavior. In both scenarios, real datasets as well as synthetic datasets are used to evaluate performance of the proposed approaches.

In the fourth part, we focus on efficiently investigating the target agent's past transactions to detect his dynamic behavior. In Chapter 7, we study the target agent's long term past behavior to find out relevant patterns for guiding trust assessment. In Chapter 8, we particularly study an agent's imprudence behavior when he is considered 'reliable' by traditional reputation systems.

Finally, Chapter 9 concludes this thesis with a summary of research. Future extensions to this research work are also outlined in this chapter.

Chapter 2

Background and related works

This chapter provides background knowledge in trust and reputation and reviews existing representative trust mechanisms in distributed systems.

2.1 Trust management

Trust is an important abstraction used in diverse scenarios including e-commerce, distributed and Peer-to-Peer systems, grid systems and dynamic collaborative systems. However, it is difficult to give a precise definition of trust because it manifests itself in many different forms and it is being used with a variety of meanings [65]. While an in-depth discussion of trust in various areas is beyond the scope of this thesis, Section 2.1.1 and 2.1.2 summarize definitions of trust and reputation to provide sufficient background for the purpose of understanding various aspects of trust and reputation. We also review existing trust and reputation systems designed for distributed systems in Section 2.2.

2.1.1 Definition of trust

According to Gambetta [66]’s definition, “*Trust (or, symmetrically, distrust) is a particular level of the subjective probability with which an agent assesses that another agent or group of agents will perform a particular action, both before he can monitor such action (or independently of his capacity ever to be able to monitor or enforce it) and in a context in which it affects his own action*”. From this definition we can see that trust

is the relationship between two entities which can be individual agents or a group of agents. Trust is subjective, because it is estimated from perspective of the individual trustor. Furthermore, trust exists before the action is monitored. That is to say, trust is a prior belief about an agent's behavior. Another point needs to be noticed is that trust is in a certain context, which affects trustor's own actions.

Marsh [67] was one of the first to define trust from a computational point of view. He agrees that trust implies some degree of uncertainty and hopefulness or optimism regarding an outcome, being subjective and dependent on the views of the individual [68]. The author categorized trust in three types as following:

- (1) Basic trust, which models the general trusting disposition independently of whoever is the encountered agent. This trust is actually the accumulated direct experience. Positive direct experience leads to a greater disposition to trust, and vice versa.
- (2) General trust, which is the trust that one agent has in another agent without considering any specific situation.
- (3) Situational trust, which is the trust that one agent has in another taking into account a specific situation.

Grandison and Sloman [69] gave a definition of trust: “*the firm belief in the competence of an entity to act dependably, securely and reliably within a specified context*” (assuming dependability covers reliability and timeliness). The properties of trust relationships can be summarized as follows:

- There is no absolute trust relationship. A trustor will not trust an entity to perform any “arbitrary” tasks. That is, the trust is situated in a specific context.
- Trust relationships can be one-to-one, i.e., between individual entities; or one-to-many, i.e., between individual entity and members of a group of entities; or many-to-many, i.e., between members of a group; or many-to-one, i.e., between members of a group and the individual entity.

- Trust may not be symmetric. That is entity A trusts entity B does not mean B trusts A either.
- Despite criticism [70], transitivity exists in some trust relationships, and thus should be considered.
- There is often a level of trust associated with a relationship. That is, concerning a specific task, one entity may be more trusted than another entity.

Jøsong et al. [71] identified two definitions of trust: (1) *reliability trust*, which is defined according with Gambetta [66]: “*Trust is the subjective probability by which an individual, A , expects that another individual, B , performs a given action on which its welfare depends.*” and (2) *decision trust*, which is inspired by [65]: “*Trust is the extent to which one party is willing to depend on something or somebody in a given situation with a feeling of relative security, even though negative consequences are possible.*”

The definitions introduced above demonstrate various aspects of the concept of trust and are very necessary to study to comprehensively understand “*what is trust?*”. But when designing a trust management mechanism, subjective probabilistic view is more suited to a computational model. In fact, in this thesis we adapt the definition from Gambetta [66].

2.1.2 Definition of reputation

Reputation is one of the ways to build trust, where one can decide whether or not to trust another based on reputation. In [72], Abdul-Rahman and Hailes stated that “*A reputation is an expectation about an agent’s behavior based on information about or observations of its past behavior*”. In [71], reputation is defined as “*what is generally said or believed about a person’s or thing’s character or standing*”. Wikipedia¹ gives the definition of reputation as “*opinion (more technically, a social evaluation) of the public toward a person, a group of people, or an organization*”. These definitions demonstrate

---

¹<http://en.wikipedia.org/wiki/Reputation>

that reputation is derived from the underlying social network, that is, it is globally visible to, or a shared perception of all the members of the network.

Reputation connects closely to the concept of trust, but there is a clear difference, which can be illustrated by the following two scenarios:

- (1) A trusts B because B has a good reputation. This reflects that reputation can be used to build trust.
- (2) A trusts B despite B 's bad reputation. This reflects that even if A knows B 's reputation, A has its own private knowledge (e.g., direct experience about B), which is more important for A to derive B 's trustworthiness than reputation.

From these two scenarios, we notice that trust is subjective and personalized. It can be derived based on various factors, in which some of them (e.g., direct experience) are more important than others (e.g., reputation).

2.2 Trust and reputation systems for distributed environments

In this section, we focus on the trust and reputation systems designed for distributed systems. Properties of large-scale systems bring new requirements to their trust management. Besides traditional requirements in most of trust management like accuracy and computation efficiency, other design issues such as the scalability, bandwidth/storage consumption, network membership dynamics and so on must be seriously considered. Note that the notations in the following part of this chapter are only valid within the individual subsection where they are defined.

2.2.1 A Distributed Trust Model

Abdul-Rahman et al. [17] proposed a distributed trust model based on conditional transitivity of trust². Consider the scenario that A trusts B and B trusts C , one can simply derive that A trusts C if the conditions are true:

---

²Other transitive trust based approaches include [36, 38, 37, 73], etc.

- B recommends its trust in C to A explicitly;
- A trusts B as a recommender; and
- A can judge B 's recommendation and decide how it will trust C , irrespective of B 's trust in C .

The properties of a trust relationship in distributed trust model include: (1) It is always between exactly two entities, (2) It is non-symmetrical, and (3) It is conditional transitive. The authors defined two types of trust relationships, *direct trust* and *recommender trust*. The model categorizes a trust relationship between entities in terms of different interactions. Trust in one category is independent of trust in other categories. For instance, A needs a car service from D who is unknown to A . A trusts B 's recommendation and asks him about D in car service category. B does not know D either so he requests C , which has D 's trust information in terms of car service. C sends D 's trust value to A . In this way recommendation path is formed. When multiple recommendation paths exist, the final trust is average of all the paths.

This trust model has several drawbacks:

- It does not explicitly address false recommendations, which significantly influence the final trust value;
- It does not provide efficient mechanisms to update trust values, which may change frequently in dynamic distributed environments.
- It is non-trivial to explore a recommendation path in large-scale distributed system.

2.2.2 EigenTrust

EigenTrust [25] is a reputation system developed for Peer-to-Peer networks. It tries to fulfill the goals of *self-policing*, *no profit for newcomers*, *anonymity-maintaining*, *minimal*

overhead and *robust to malicious collectives of peers*. EigenTrust relies on transitivity of trust and adapts the ideas from PageRank algorithm [74].

EigenTrust calculates a global reputation for each peer in the network based on the local opinions of all other peers. Every peer i stores for every interaction partner j a trust value s_{ij} , representing the experience it has gained. s_{ij} is normalized locally (denoted by c_{ij}) to avoid the arbitrary values assigned by malicious peers. Peer i then asks its acquaintance what they think about other peers and receives the normalized trust values. All the normalized values are stored in a matrix. From this matrix, peer i can retrieve any peer j 's global reputation t_{ij} by calculating

$$t_{ij} = \sum_k c_{ik} \cdot c_{kj} \quad (\text{Eq. 2.1})$$

So peer's global reputation is the weighted sum of all other peers' opinions where the weight factor is global reputation of the opinion reporter.

Let C be the matrix $[c_{ij}]$, define $\vec{t}_i$ and $\vec{c}_i$ to be vectors that contain values t_{ij} and c_{ij} respectively. Then the global reputation is $\vec{t}_i = (C^T)\vec{c}_i$. To broaden the view further, a peer will ask its friends' friends' opinions: $((C^T)(C^T))$. Increasing n in $\vec{t}_i = (C^T)^n \vec{c}_i$ continues to broaden the view. Finally, $\vec{t}_i$ will converge to the same vector (i.e., all peers will converge to the same values representing global reputation).

EigenTrust uses distributed hash table (DHT) to store and lookup global reputation scores. A score manager is responsible for calculating, storing and communicating reputation scores for the identities who fall in the score manager's responsibility range in the DHT. The use of multiple hash functions allows multiple score managers for each host to provide redundancy.

The calculation and reputation scores dissemination is not complex thus the system only needs a minimal overhead. The use of DHT in EigenTrust makes it eligible only for structured P2P systems and is subject to drawbacks of DHT [75]. The main problem of EigenTrust algorithm is the assumption of pre-trusted peers [28]. Firstly, such peers are hot spots for any kind of attack. Secondly, even if these pre-trusted peers can defend

various attacks, there are no mechanisms to guarantee that they are trustworthy forever. Furthermore, it does not always make sense to use global reputation of opinions reporter as the weight of the corresponding opinion [26].

2.2.3 PeerTrust

Evolution of Trustworthiness of the peers in PeerTrust [26] is based on five factors: (1) feedback that a peer obtains from other peers, (2) feedback scope like number of past transactions, (3) credibility of feedback provider, (4) transaction context factor and (5) community context.

Feedbacks received from the network are weighted by credibility of the feedback provider. To assess the credibility, authors first used the trust value of the feedback provider as the credibility factor so feedbacks from the trustworthy peers are considered more credible. Such strategy is based on the assumptions that trustworthy peers are more likely to provide correct feedbacks and untrustworthy peers are more likely to provide false feedbacks. However, these assumptions are not always true in the real world. For instance, a malicious peer may submit genuine feedback aperiodically to avoid detection while a honest peer who provides high quality service may send false feedbacks about its competitors. The authors thus proposed a similarity based approach. To measure the credibility of peer j , peer i computes the similarity between them by comparing their opinions about the peers that both of them have interacted with. More weight is added to the feedback provided by the peer who has the similar personalized experience with requestor.

The transaction context factor can be a timing function, where recent transactions have a greater impact. The intention of community context factor is to convince peers to give feedbacks of past transactions. Authors gave the example as the ratio of number of feedbacks a peer submitted over the number of transactions this peer has. Obviously, submitting more feedbacks helps increase a peer's trust. Equation Eq. 2.2 shows how

these factors are combined to present a general trust metric T for peer u .

$$T(u) = \alpha \cdot \sum_{i=1}^{I(u)} S(u, i) \cdot Cr(P(u, j)) \cdot TF(u, i) + \beta \cdot CF(u) \quad (\text{Eq. 2.2})$$

Where $I(u)$ denotes the number of transactions of peer u with other peers, and $p(u, i)$ denotes the interaction partner (peer) in peer u 's i th transaction. $S(u, i)$ is the normalized amount of positive ratings u receives from $p(u, i)$ in i th transaction, $Cr(v)$ denotes the credibility of the feedback provided by peer v , $TF(u, i)$ is the adaptive transaction factor for u 's i th transaction, and $CF(u)$ is the adaptive community context factor for peer u . α and β are the normalized weights for the collective evaluation and the community context factor.

The main contribution of this work is to design a comprehensive framework covering most of factors that affect reputation of a peer. The main problem of this work is that the computation convergence rate in large-scale P2P systems is not provided. Furthermore, the five factors used in this model incur large overhead, which has been shown/argued in the literature [28, 76].

2.2.4 P2PRep

P2PRep [33] is designed to mitigate the effects of malicious and selfish peers in an anonymous P2P systems. Reputation is viewed at two levels in the system: (i) local reputation (based on direct interactions) and (ii) network reputation (based on indirect interactions). Boolean value is used to measure the outcome of an interaction. 1 indicates the outcome is satisfactory and 0 otherwise. When a peer requests a kind of service, a collection of potential service providers are returned. If there is sufficient local experience, this peer computes the reputation score by

$$r_{ij}^{(n)} = \begin{cases} t_{ij}^n & \text{if } n = 1 \\ \alpha^{(n)} r_{ij}^{(n-1)} + (1 - \alpha^{(n)}) t_{ij}^{(n)} & \text{if } n \geq 2 \end{cases}$$

Where t_{ij}^n represents the outcome of the n th interaction between peer i and j . $r_{ij}^{(n)}$ represents j 's local reputation calculated by i after n interactions. $\alpha^{(n)} \in [0, 1]$ is the

aggregation freshness indicating the importance of past interactions in relation with the last one. Freshness is adjusted by checking whether the current reputation score can predict the interaction partner's next behavior accurately. If current score is reliable, then α does not change, otherwise, α is changed accordingly. The accuracy of a reputation score is a binary value and is computed by

$$Acc_{ij}^{(n)} = \begin{cases} 1 & \text{if } |r_{ij}^{n-1} - t_{i,j}^n| < E \\ 0 & \text{otherwise} \end{cases}$$

Where E is the error threshold. $\alpha^{(n)}$ is calculated by $\beta^{(n)}/2$, where $\beta^{(n)}$ is a coefficient and is updated by

$$\beta^{(n)} = \frac{\beta^{(n-1)} + Acc_{ij}^{(n)}}{2} \quad (\text{Eq. 2.3})$$

If there is no (mature)local experience, the service requestor polls the network about reputation of the potential service providers. Before aggregating these feedbacks, service requestor will choose a subset of feedbacks to verify by polling other peers about the reputation information about the feedback reporter (i.e., based on voting). This can help reduce the false feedbacks. Using an ordered weighted average, the feedbacks obtained from the network are combined with the local score (if any) to generate a final reputation score.

P2PRep uses public key cryptography to maintain user anonymity and packet integrity. All the replies are signed using requestor's public key, protecting both data integrity and responder's identity. Only requestor can decrypt the message. One drawback of this work is that it does not preserve the anonymity of peers during the voting phase of protocol (i.e., IP address is revealed) [77]. Furthermore, the authors did not explicitly consider the churn in the networks, which may affect the feedback gathering and verification processes [78].

2.2.5 PowerTrust

PowerTrust [28] aims to fulfill the key design issues of *high accuracy, fast convergence speed, low overhead, adaptive to peer dynamics, robust to malicious peers* and *scalability* for trusted Peer-to-Peer computing.

The authors proposed a Trust Overlay Network (TON) built on top of P2P network, where any pair of interacting peers are connected with feedback score. When a new interaction occurs between a pair of peers, feedback score is updated accordingly. Each peer i stores local reputation scores of all the peers it has interacted. Let s_{ij} be the raw trust score i assigns to j , the corresponding local reputation score r_{ij} is normalized as $s_{ij} / \sum_k s_{ik}$. All the local reputation scores are stored in a matrix $R = (r_{ij})$.

Global reputation score v_i of peer i is computed based on other peers' local reputation score. All the v_i are stored in a normalized reputation vector $V = (v_i)$, where $\sum_i v_i = 1$. V is updated recursively by

$$V_{(t+1)} = R^T \cdot V_{(t)} \quad (\text{Eq. 2.4})$$

Computation reaches convergence when $|V_{(t+1)} - V_{(t)}| < \epsilon$, where ϵ is the error threshold. This process is very similar to EigenTrust [25]. To improve aggregation efficiency, authors proposed *look-ahead random walk* (LRW) strategy where each peer combines its local reputation scores and neighbors' first hand ones. The enhanced trust matrix $S = R^2$.

To select the most reputable peers dynamically. PowerTrust uses DHT with locality preserving hash function (LPH). Every peer has a score manager. If score manager j is responsible for peer i , it stores the tuple (v_i, i) . j hashes v_i using LPH. j inserts the tuple (v_i, i, j) to the successor peer of the hashed value. When a peer wants to find the m most reputable peers, it asks the peer who is responsible for the highest hash value. If this peer stores m highest rated peers, it returns those, otherwise, it asks its predecessor and so on.

When aggregating the initial global reputation, a score manager who is responsible for peer j receives all the messages (r_{ij}, i) . For every peer i , it retrieves its global reputation v_i and calculates:

$$v_{j(t+1)} = v_{j(t)} + v_i \cdot r_{ij} \quad (\text{Eq. 2.5})$$

The process is repeated until $|v_{j(t+1)} - v_{j(t)}| < \epsilon$, where ϵ is the error threshold.

Update of reputation follows a random walk through TON. A greedy factor is defined as the eagerness probability of the surfer jump directly to the power nodes. So power nodes are updated more often to avoid that one of them starts to subvert the system.

PowerTrust simplifies the calculation in comparison with EigenTrust. But PowerTrust relies on two overlays, one for peers with their score managers and another for ranking peers according to their global reputation. So like EigenTrust, such mechanism is only applied to structured P2P systems and its performance is influenced by underlying DHT greatly.

2.2.6 Beta reputation system and TRAVOS

The beta reputation system proposed by Jøsang et al. [19] estimates reputation of an agent using a probabilistic model, i.e., based on the beta probability density function (PDF)³. The beta distributions are a family of statistical distribution functions that are characterized by two parameters α and β . The beta probability density function is defined as follows:

$$beta(p|\alpha, \beta) = \frac{\Gamma(\alpha + \beta)}{\Gamma(\alpha)\Gamma(\beta)} p^{\alpha-1} (1 - p)^{\beta-1} \quad (\text{Eq. 2.6})$$

Where $p \in [0, 1]$ is a probability variable, $\alpha, \beta > 0$. This function shows the relative likelihood of the values for p , given the parameters α and β . The probability expectation of the beta distribution computed by:

$$E(p) = \frac{\alpha}{\alpha + \beta} \quad (\text{Eq. 2.7})$$

In the beginning, when knowledge is not available, the priori distribution is actually the uniform beta PDF with the parameters $\alpha = 1$ and $\beta = 1$ (see Fig. 2.1.a). After observing x positive and y negative outcomes, the posterior distribution becomes beta PDF with $\alpha = 1 + x$ and $\beta = 1 + y$. Fig. 2.1.b gives the example when $x = 10$ and $y = 2$.

---

³Note that although the beta reputation system is described as a centralized approach in the original paper, its beta distribution based model has been widely used in distributed settings [79, 80, 44].

2.1.a: Beta PDF with $\alpha = 1$ and $\beta = 1$. 2.1.b: Beta PDF with $\alpha = 11$ and $\beta = 3$

Figure 2.1: Examples of beta PDF.

Following this model, the feedbacks collected from other agents who have interacted with the target agent are binary (1 or 0), i.e., 1 means that the target agent has a good reputation and 0 otherwise. These feedbacks are combined by simply accumulating the number (a) of positive feedbacks and the number (b) of negative feedbacks. Hence, To make $\alpha, \beta > 0$, their values are set as:

$$\alpha = 1 + a, \beta = 1 + b \quad (\text{Eq. 2.8})$$

The posteriori reputation score is computed by combining a prior reputation score with the new feedbacks.

In beta reputation system, the authors presented a basic model to compute/update the target agent's reputation following beta distribution. However, since there are no effective mechanisms provided to filter out unfair, this work is only effective when most of the feedbacks are genuine.

TRAVOS [80] is a trust and reputation model for agent-based virtual organizations. This work also makes use of beta distribution to compute trust but it pays more attention to the issue of unfair feedbacks. When the trustor evaluates a potential interaction partner, he first uses his own local knowledge to derive trust of the target agent. He then estimates accuracy of the direct trust, that is the probability that the real likelihood of cheating falls in the certain range from the trustor's estimation. If the direct trust is accurate enough, the trustor simply relies on this direct experience based trust. Otherwise, he requests feedbacks from other agents who have interacted with the target agent.

To ensure that only accurate feedbacks are considered, TRAVOS addresses inaccurate reputation feedbacks by performing two tasks:

- Estimating the probability that a feedback provider's opinion of the target agent is accurate. This is done by comparing current feedback with the previous feedbacks provided by the same agent. The accuracy of the current feedback is the expected value of the beta PDF representing the numbers of successful and unsuccessful interactions between the trustor and the target agent when the trustor is guided by the previous feedbacks.
- Adjusting reputation feedback according to its accuracy to reduce the effect of the inaccurate feedback.

Compared to the beta reputation system [19], TRAVOS (as well as other beta distribution based enhancements [23, 27, 81], etc.) takes into account the issue of unfair feedbacks thus deriving more accurate reputation score of the target agent. However, TRAVOS assumes that an agent acts consistently, which is clearly not true in reality. For instance, in Internet auction sites, malicious sellers may first act honestly in some transactions selling cheap items and then cheat in a transaction selling expensive items. Moreover, both beta reputation system and TRAVOS rely on an agent's past behavior (direct or indirect experience), which is not always available. For instance, an agent is new to the system; or an agent interacts with others infrequently.

In advance of TRAVOS, Zhang et al. [20] took into account an agent's dynamic behavior by introducing the concept of time window. That is, the ratings of the target agent are partitioned into different elemental time windows. In each time window, the trustor counts the numbers of successful and unsuccessful transactions. The trustworthiness of the target agent is firstly calculated by aggregating numbers of successful and unsuccessful transactions in each time window (taking into account forgetting rate) and then is adjusted according to reputations of the indirect experience providers.

2.2.7 Stereotyping based approaches

Using stereotypes to model behavior of interaction partner does not appear for the first time and has been studied in the areas of user modeling [82, 83] and recommender systems [84, 85]. Nevertheless, StereoTrust is the first concrete stereotyping based trust model. After StereoTrust, several other stereotyping based works are proposed in the area of multi-agent system. Burnett et al. [86] used stereotypes to address the cold-start issue. The trustor constructs stereotypes relying on machine learning algorithm (i.e., M5 model tree learning [87]), and combine the stereotypes with the target agent's reputation (if any) using subjective logic [88]. Furthermore, the authors extended the model by allowing new trustors to request experienced trustors who have already formed local stereotypes. The proposed MetaTrust framework (see Chapter 4) is very similar to this work, but it has several advantages: (i) MetaTrust provides a comprehensive trust framework where appropriate machine learning algorithms are automatically selected and applied to the specific application scenarios while this work lacks such functionality; (ii) this work shares agents' local stereotypes in an ad-hoc manner while MetaTrust offers a more sophisticated sharing mechanism which maintains a dedicated overlay network for exchanging, combining, updating stereotypes, as well as preserving agents' privacy; (iii) unlike this work which used a simulated agent society for evaluation, MetaTrust is validated on a real auction dataset thus is more realistic.

Burnett et al. [89] looked at the different aspect of the problem, that is to identify useful features to construct stereotypes. Three feature sources are discussed. (i) From social network. That is, relationships between agents can be viewed as features. For instance, such relationships could be *agent A is a friend of agent B*. (ii) From agents' competence over time. The target agent's accumulated experience in certain tasks can be viewed as features. An example stereotype may be *if an agent performed task T for over 100 times, he is considered experienced (trustworthy)*. (iii) From interactions, e.g., features of both interaction parties. For instance, the trustor with certain features is positively or negatively biased towards the target agent with other features. This work

provided a comprehensive summary of feature sources (for stereotype formation) from social relationships among agents but the authors did not apply these features to any concrete application scenarios for validation.

Part II

Trust based on stereotyping

Chapter 3

StereoTrust: a stereotypes based personalized trust model

In order to remedy flaws of the existing reputation based trust mechanisms discussed in Chapter 1, we wish to develop novel trust models that assess trustworthiness without the target agent’s historical information. To that end, we use a different kind of information to build stereotypes, which are used to aid in trust assessment, inspired by real life human interactions: in order to make the first guess about the trustworthiness of a stranger, we commonly use our “instinct” – essentially stereotypes developed from our past interactions with “similar” other people. A stereotype contains certain features of agents and an expected outcome of the transaction. These features can be taken from the agents’ profile information (age, location, role, etc.), or other behavioral clues (category, gain, risk, etc. of the transaction) that may be observed about the target. When facing a stranger, the stereotypes matching a stranger’s profile are aggregated to derive an expected trust. Figure 3.1 depicts and contrasts the traditional computation flow versus the proposed StereoTrust [44]’s computation flow for trust assessment. Additionally, when some information about a stranger’s previous transactions is available, it may in addition be used to refine the stereotyping based prediction. Note that although stereotypes have already been used in other areas (e.g., user modeling [83], recommender systems [84], etc.), StereoTrust is the first concrete stereotyping based trust model.

This chapter is structured as follows: Section 3.1 outlines the basic idea and working examples of the proposed StereoTrust model. The StereoTrust model and historical

Figure 3.1: Trust computation flow: reputation based trust mechanisms versus StereoTrust

information based possible enhancements are elaborated in Section 3.2. Validation using real Epinions dataset as well as synthetic data is presented in Section 3.3. Section 3.4 summarizes this chapter.

3.1 Introduction

StereoTrust is a novel computational trust model in which the target agents' trust is estimated using stereotypes learned by the assessor from his (the assessor's) own interactions with other agents with profile similar to that of the target agent, where the profile is determined based on features associated with the agent/transaction. An analogy can be drawn between StereoTrust and [90, 91], which study the relation between the reputation of a company and its employees: The company's reputation can be modeled as an aggregate of the employees' reputations and it can suggest a prior estimate for an employee's reputation. In StereoTrust, when facing an unknown target agent, trustor forms stereotypes using two kinds of information: (1) interaction experience (i.e., past transactions) with other agents, (2) features that can be collected from agents' profiles or context of the transactions. An example of stereotype is: When a trustor needs to evaluate programming skills of an unknown programmer. Unlike traditional methods which rely on inputs from others who directly or indirectly know the target (such in-

formation is often unavailable or inaccurate), the trustor first identifies some features which may correlate with programming skills such as reputation of the company that hires the programmer, the programmer’s highest related degree and the university where the degree is awarded, etc. Based on these features, trustor aggregates his experience with other (known) programmers who have the same features (e.g., the same company, degree or university) with the target and estimates target’s programming skills based on such aggregated experience, e.g., programmers employed by a reputable software company are more skilled than average.

StereoTrust has its own weaknesses and limitations. For instance, not in all circumstances may a user determine the profile of an agent. However, we think that StereoTrust is interesting both academically and in practice. Academically, its novelty lies in emulating a real world human behavior for modeling trust by using stereotypes for a first guess about a stranger. In practice, firstly, StereoTrust may be applicable under certain circumstances even when information used by traditional trust models is not available, or noisy, inaccurate or tampered and hence unreliable; secondly, StereoTrust facilitates personalized recommendations and finally, StereoTrust can be seamlessly extended to incorporate and leverage on global information used by traditional trust models, if available, as well, in order to enhance the prediction (thus, the term “stereo” in the model’s name is a double entendre). Our experiments show that such a refinement, called d-StereoTrust, significantly improves the accuracy. Please note that original StereoTrust differs traditional trust models by making use of different information inputs, i.e., experience aggregated and characterized from other *similar* agents. Utilizing global information (inputs of conventional approaches) is optional (d-StereoTrust), conditional (necessary information is available and accurate), and is for the purpose of prediction accuracy improvement, so this does not mix up the principles of StereoTrust and traditional trust models.

StereoTrust is generic, and its use of abstract group definitions allow it to be used

in very different kinds of applications¹, and even within a same application, different agents may have their own personal, locally defined groups. Also, the notion of trust itself can be easily adapted to different contexts.

In this thesis, we adopt the definition of trust from [66]: “*Trust (or, symmetrically, distrust) is a particular level of the subjective probability with which an agent assesses that another agent or group of agents will perform a particular action, both before he can monitor such action (or independently of his capacity ever to be able to monitor or enforce it) and in a context in which it affects his own action*”.

For an example application, that of a Peer-to-Peer storage system. If a peer wants to store a new block of data at some peers, it would need to choose a suitable peer to do so. The suitability of a peer may depend on the likelihood that the peer will be available when the data needs to be retrieved (which may depend on its geographic location/time-zone difference), the response time to access the data (which may depend on agreements and infrastructure between ISPs, but the user would need to perceive it by measurements), and so on. Conventional systems design approach models such a scenario as a multi-criterion optimization problem. Such an approach would typically need knowledge about the specific peer in question - for instance, what time does this peer come online and go offline, what is the end to end latency and available bandwidth (which in practice varies over time, and is hard to measure and anticipate), etc. Applying StereoTrust can provide an alternative systems design², where a peer in, say Tokyo, can think - “my past experiences tell me that peers in Beijing and Hong Kong have more common online time with me compared with peers in London and New York. Likewise, peers in New York and Hong Kong with a specific IP prefix provide reliable and fast connections, while the others don’t.” Based on such information, the peer would be able to make a first guess that a peer in Hong Kong is likely to be its best bet, if it has to

---

¹See Part III where we deploy StereoTrust for two diverse applications, namely P2P storage and online auction.

²We are not claiming that it necessarily provides the best possible system design, but merely that it opens the opportunity for alternative designs. We have in fact devised such StereoTrust guided data placement strategy for P2P backup systems, resulting in several desirable properties w.r.to other existing placement strategies [92].

choose between a peer in Hong Kong and London, without needing to know the history of the specific peers in question. A ‘mixed’ data placement strategy, using historical information, when available, in addition to using such stereotypes, is expected in turn, to result in even better performance.

3.2 Description of stereotyping based Trust Model

We refer to a participant in the system as an agent. We denote by $\mathcal{A}$ the set of all agents in the system; and by $\mathcal{A}_x$ the set of agents known to agent a_x . An agent can provide services for other agents. A transaction in the system happens when an agent accepts another agent’s service. To indicate the quality of a service, an agent can rank the transaction. Θ_{a_x, a_y} denotes the set of transactions between service provision agent a_y and service consumption agent a_x . $\theta_{a_x, a_y} = |\Theta_{a_x, a_y}|$ denotes the number of such transactions.

3.2.1 Group Definition

In StereoTrust, a group is a (implicit) community of agents who show some common properties or behave similarly in certain aspects, from the perspective of the trustor. Thus, this group may be universal in nature (e.g., graduates from NTU Singapore), or subjective, determined by the perspective of the trustor (e.g., people from the neighboring countries). Likewise, the opinion about the group itself is also subjective, based on the trustor’s own experiences (or beliefs). Because of the common properties shared by all the members of this group, we believe that the group can act as a collective entity to represent its member agents (to a certain extent). For instance, people may consider a programmer working in a well-known software company as skilled, even if they do not personally know the person. People trust the company based on the quality of produced software, thus they also trust the programmers who create the software. On the other hand, a company employing skilled (i.e., trusted) programmers can release high-quality products, and thus gain high reputation. Such interplay between a group

and its members' reputation is the basis of our work. An assessor derives the trust of an agent according to his own notion of groups, and the trust he has on the corresponding groups.

Groups are defined subjectively by the agent a_x that uses StereoTrust to derive trust on other agents. A group G_x^i is a set of agents. We denote by $\mathcal{G}_x = \{G_x^1, G_x^2, \dots, G_x^n\}$ the set of all groups defined by a_x . Based on a_x 's previous experience, stereotypes, and any other information, a_x formulates grouping functions $M_x(G_x^i, a) : [\mathcal{G}_x, \mathcal{A}_x] \rightarrow [0, 1]$, which, for each group G_x^i , map agent a to the probability that a is the member of this group. Thus, in the most general model, a group is a fuzzy set of agents. If $M_x(G_x^i, a) = 1$, it is certain that a is member of G_x^i ($a \in G_x^i$); if $M_x(G_x^i, a) = 0$, it is certain that a does not belong to G_x^i ($a \notin G_x^i$).

Note that in this chapter we do not discuss how to derive the grouping functions $M_x(G_x^i, a)$. However, during our experiments, we propose how to formulate such functions for Epinions.com dataset. For instance, a group can gather all agents responsible for the same type of task; or all agents interested in a certain topic; or all agents living in the same location. Depending on the type of criteria used, groups may overlap (an agent belongs to multiple groups simultaneously) or be disjointed (each agent belongs only to one group).

3.2.2 Modeling Trust

A computational trust model represents the complex notion of trust by a variable, which can be binary, discrete, continuous, etc. We assume, in lines with a standard assumption [66] in the literature, that trust indicates the probability that an agent will perform a particular, expected action during a transaction. Thus, agent's trust rating is a real number from the range $[0, 1]$, where 0 indicates that the agent is absolutely untrustworthy and 1 indicates that the agent is absolutely trustworthy.

The beta distribution is commonly used to model uncertainty about probability p of a random event and has been used to model an agent's reputation [19, 79]. A series

of transactions between a pair of agents are modeled as observations of independent Bernoulli trials. In each trial, the success probability p is modeled by beta distribution with parameters α and β (starting with $\alpha = \beta = 1$, that translate into complete uncertainty about the distribution of the parameter, modeled by the uniform distribution: $Beta(1, 1) = U(0, 1)$). After observing s successes in n trials, the posterior density of p is $Beta(\alpha + s, \beta + n - s)$ [41].

Trust function between *entities* (an individual agent or a group of agents) may be defined based on a beta function. By E_t , we denote the entity participating in the trust calculation.

Definition 3.1 (Trust Function) *Entity E_1 evaluates entity E_2 . From the viewpoint of E_1 , S_{E_1, E_2} and U_{E_1, E_2} represent, respectively, the number of successful transactions and unsuccessful transactions between E_1 and E_2 ($S_{E_1, E_2} \geq 0$ and $U_{E_1, E_2} \geq 0$). Trust function $T_{E_1, E_2}(p|S_{E_1, E_2}, U_{E_1, E_2})$ mapping trust rating p ($0 \leq p \leq 1$) to its probability is defined by:*

$$T_{E_1, E_2}(p|S_{E_1, E_2}, U_{E_1, E_2}) = \frac{\Gamma(S_{E_1, E_2} + U_{E_1, E_2} + 2)}{\Gamma(S_{E_1, E_2} + 1)\Gamma(U_{E_1, E_2} + 1)} \cdot p^{S_{E_1, E_2}}(1 - p)^{U_{E_1, E_2}}. \quad (\text{Eq. 3.1})$$

The expected value of the trust function is equal to:

$$E_{E_1, E_2}(T_{E_1, E_2}(p|S_{E_1, E_2}, U_{E_1, E_2})) = \frac{(S_{E_1, E_2} + 1)}{(S_{E_1, E_2} + U_{E_1, E_2} + 2)} \quad (\text{Eq. 3.2})$$

3.2.3 StereoTrust Model

StereoTrust model uses only information locally available to an agent, for him to derive another agent's trust. Consider a scenario where an agent a_x , a service requestor, encounters a potential service provider a_y with whom a_x had no prior experience. We assume that a_x can obtain some features about a_y , such as a_y 's interests, location, age, etc. Such information as well as other information like a_x 's previous experience is used by a_x to form groups that help derive a_y 's trust.

In the basic model, StereoTrust considers only groups for which the membership is certain. We denote these groups by $\mathcal{G}_{x,y} = \{G_{x,y}^1, G_{x,y}^2, \dots\}$ such that $M_x(G_{x,y}^i, a_y) = 1$ (for the sake of simplifying the notation, we will use G^i in place of $G_{x,y}^i$ when the context is clear). The trust between a_x and each of these groups G^i is derived based on past interactions with agents that belong to G^i with certainty. Thus, from the set of all agents a_x has previously interacted with ($\mathcal{A}_x = \{a_1, a_2, \dots\}$), a_x extracts those that belong to G^i (i.e., $G^i = \{a : M_x(G^i, a) = 1\}$). Then, a_x counts the total number of successful S_{a_x, G^i} transactions with G^i by summing up the successful transactions with G^i 's members: $S_{a_x, G^i} = \sum_{a \in G^i} S_{a_x, a}$. The total number of unsuccessful transactions U_{a_x, G^i} is computed similarly. Finally, a_x uses Eq. 3.1 to derive G^i trust function.

To derive agent a_y 's trust value, a_x combines his trust towards all the groups $\mathcal{G}_x$ in which a_y is a member. The trust is computed as a weighted sum of groups' trust with weights proportional to the fraction of transactions with that group. For group G^i , weight factor $W_{x,y}^i$ is calculated as a number $\theta_{x,y}^i$ of a_x 's transactions with $G_{x,y}^i$ members ($\theta_{x,y}^i = |\Theta_{a_x, a}|$ such that $a \in G_{x,y}^i$); divided by the total number of a_x 's transactions with members of any $\mathcal{G}_{x,y}$ group. Obviously, the higher the number of transactions regarding one group, the more likely is a_x to interact with agents of this group, so this group contributes more to represent a_y 's trust from viewpoint of a_x . We define weight factor $W_{x,y}^i$ for G_i as:

$$W_{x,y}^i = \frac{\theta_{x,y}^i}{\sum_j \theta_{x,y}^j} \quad (\text{Eq. 3.3})$$

Using the estimated weights, we combine all group trusts to derive a_y 's trust. The process of trust calculation is illustrated in Figure 3.2. We propose two approaches to calculate and combine group trusts:

SOF Approach (Sum Of Functions)

In this approach, we first calculate probability density of trust rating for each group using trust function (Eq. 3.1) and then combine them to produce a_y 's probability density of trust rating $TD_{a_x, a_y}(p)$ using Eq. 3.3:

$$TD_{a_x, a_y}(p) = \sum_i W_{x,y}^i \cdot T_{a_x, G^i}(p | S_{a_x, G^i}, U_{a_x, G^i}), \quad (\text{Eq. 3.4})$$

Figure 3.2: Process of trust calculation. Weighted sum of each group G^i 's trust by assigning corresponding weight factor $W_{x,y}^i$

where S_{a_x, G^i} and U_{a_x, G^i} are aggregated numbers of successful and unsuccessful transactions between a_x and members of group G^i .

SOP Approach (Sum Of Parameters)

In this approach, we use only one trust function by setting the parameters, i.e., numbers of corresponding successful and unsuccessful transactions.

$$TD_{a_x, a_y}(p) = T_{a_x, a_y}(p) \left| \sum_i W_{x,y}^i \cdot S_{a_x, G^i}, \sum_i W_{x,y}^i \cdot U_{a_x, G^i} \right|, \quad (\text{Eq. 3.5})$$

where S_{a_x, G^i} and U_{a_x, G^i} are defined as in SOF approach.

3.2.4 Dichotomy Based Enhanced Model

StereoTrust model simply groups agents based on their profiles. This limits StereoTrust model from accurately predicting the performance of an agent who behaves quite differently from the other agents of the same groups.

To improve prediction accuracy, we propose dichotomy-based enhancement of StereoTrust (called d-StereoTrust). The main idea is to construct subgroups that divide agents on a finer level than groups based on agents' profiles. Additional information (third party information) is needed in this case. Figure 3.3 illustrates this kind of grouping.

In d-StereoTrust, each top-level group G^i is further divided into two subgroups, an *honest* $G^{i,h}$ and a *dishonest* $G^{i,d}$ subgroup (hence dichotomy-based). a_x assigns an agent $a \in G^i$ to either subgroup by analyzing history of his transactions with a . The basic

 Figure 3.3: Structure of group G^i in d-StereoTrust

criterion we use is that if a_x has more successful than unsuccessful transactions with a , a is added to the honest subgroup $G^{i,h}$ (and, consequently, in the alternative case a is added to $G^{i,d}$). Several alternative criteria are possible, for instance, the average rating of transactions with a .

After dividing a group G^i into subgroups ($G^{i,h}$, $G^{i,d}$) and determining a_x 's trust towards the subgroups (computed as in the previous section), d-StereoTrust computes how similar is the target agent a_y to the honest and the dishonest subgroup. If a_y “seems” more honest, a_x 's trust towards aggregated G^i should reflect more a_x 's trust towards the honest subgroup $G^{i,h}$; similarly, if a_y “seems” more dishonest, the dishonest subgroup $G^{i,d}$ should have more impact on a_x 's aggregated trust towards G^i . This process is illustrated in Figure 3.4.

The closeness (between the target agent and the subgroups), which can be measured by membership $M_x(G^{i,\cdot}, a_y)$ of the target agent a_y to subgroup $G^{i,\cdot}$ (where $\cdot$ represents d or h) is based on other agents' opinions about a_y . Note that we cannot group a_y as any other agent $a \in \mathcal{A}_x$, because the grouping described above is based on a_x history with a , and, obviously, there are no previous transactions between a_x and a_y . Thus, both $M_x(G^{i,h}, a_y)$ and $M_x(G^{i,d}, a_y)$ are fuzzy (in $[0, 1]$).

Figure 3.4: Process of trust calculation. Trusts of honest subgroup $G^{i,h}$ and dishonest subgroup $G^{i,d}$ of each group G^i are firstly combined using closeness and then trusts of all groups G^i are combined using weight factor $W_{x,y}^i$ to derive the target agent's trust.

Agent a_x obtains opinions about a_y by requesting a certain metric from other agents. For instance, a_x can ask other agent a_k about the percentage (denoted by $m_{k,y}$) of successful transactions he had with a_y . a_x will seek opinions from honest agents from the group $G^{i,h}$; and also from agents interested in G^i , but with no transactions with a_x (based on their profile information, these agents could be classified as members of G^i , but they had no transactions with a_x). Obviously, the agents who have no transactions with a_x may be dishonest thus may provide false reports.

It is worth mentioning that the amount of historical information needed from other agents in d-StereoTrust is a small subset of information required in models based on feedbacks or transitive/Eigentrust. To collect feedbacks or form transitive trust paths, Eigentrust-like algorithms must explore the whole network (take into account all the available historic transactions). In contrast, in d-StereoTrust, a_x only asks the agents who are interested in the corresponding groups.

Based on all opinions $m_{k,y}$ received, a_x computes an aggregated opinion m_y , which is used to measure the closeness of a_y to subgroups as a simple average of $m_{k,y}$.

To characterize subgroups in a similar way, a_x computes similar aggregation of his opinions towards subgroups $G^{i,h}$ and $G^{i,d}$. Aggregated opinion m_h about subgroup $G^{i,h}$ is equal to the simple average of $m_{x,j}$ (percentage of successful transactions a_x had with

a_j), where j is the index of agent $a_j \in G^{i,h}$. The aggregated opinion m_d about subgroup $G^{i,d}$ is derived in the same way.

Finally, the closeness between a_y and each of the subgroups is computed as the fraction of the distance between m_y from one side and m_h and m_d from the other:

$$M_x(G^{i,h}, a_y) = \frac{1/(|m_y - m_h|)}{1/(|m_y - m_h|) + 1/(|m_y - m_d|)} \quad (\text{Eq. 3.6})$$

$$M_x(G^{i,d}, a_y) = \frac{1/(|m_y - m_d|)}{1/(|m_y - m_h|) + 1/(|m_y - m_d|)} \quad (\text{Eq. 3.7})$$

This procedure has a straightforward interpretation. If other agents have similar opinions about target agent a_y as a_x has about the dishonest subgroup, then the target agent is most likely dishonest, so the dishonest subgroup trust should more influence a_y 's trust in the context of group G^i . Similarly, if other agents have experienced similar performance with a_y as a_x with the honest group, then a_y is most likely honest.

Note that we do not use the opinions provided by other entities to directly calculate a_y 's trust. Instead, we use them as metrics to measure closeness between a_y and the subgroups. In other words, we do not ask other agents "is a_y honest?", but rather we ask about quantified experience they had with a_y . This allows us, firstly, to be more objective; and, secondly, to easily extend d-StereoTrust to use multiple metrics (and to combine them with, e.g., Euclidean distance). Simulation results show that d-StereoTrust derived trust is more accurate than that is derived directly using others' opinions. When other agents' opinions are not available, d-StereoTrust model degrades to StereoTrust model.

After calculating closeness, we combine groups' trusts to derive a_y 's trust. Similarly to the original StereoTrust, there are two approaches to combine various trust sources.

SOF Approach (Sum Of Functions)

Using Eq. 3.1, Eq. 3.3, Eq. 3.6 and Eq. 3.7 we have probability density of the target agent (a_y)'s trust rating $TD_{a_x, a_y}(x)$:

$$TD_{a_x, a_y}(p) = \sum_i W_{x,y}^i \cdot (M_x(G^{i,h}, a_y) \cdot T_{a_x, G^{i,h}}(p|S_{a_x, G^{i,h}}, U_{a_x, G^{i,h}}) + M_x(G^{i,d}, a_y) \cdot T_{a_x, G^{i,d}}(p|S_{a_x, G^{i,d}}, U_{a_x, G^{i,d}})), \quad (\text{Eq. 3.8})$$

Where $S_{a_x, G^{i,h}}/S_{a_x, G^{i,d}}$ and $U_{a_x, G^{i,h}}/U_{a_x, G^{i,d}}$ are aggregated numbers of successful and unsuccessful transactions of each member of G^i 's subgroup $G^{i,h}/G^{i,d}$ from viewpoint of agent a_x .

SOP Approach (Sum Of Parameters)

Using Eq. 3.1, Eq. 3.3, Eq. 3.6 and Eq. 3.7 we have probability density of agent a_y 's trust rating $TD_{a_x, a_y}(x)$:

$$TD_{a_x, a_y}(p) = T_{a_x, a_y}(p| \sum_i W_{x,y}^i \cdot (M_x(G^{i,h}, a_y) \cdot S_{a_x, G^{i,h}} + M_x(G^{i,d}, a_y) \cdot S_{a_x, G^{i,d}}), \sum_i W_{x,y}^i \cdot (M_x(G^{i,h}, a_y) \cdot U_{a_x, G^{i,h}} + M_x(G^{i,d}, a_y) \cdot U_{a_x, G^{i,d}})), \quad (\text{Eq. 3.9})$$

Where $S_{a_x, G^{i,h}}/S_{a_x, G^{i,d}}$ and $U_{a_x, G^{i,h}}/U_{a_x, G^{i,d}}$ have the same meanings with that in SOF approach.

3.3 Evaluation

In this section, we conduct experiments to evaluate the performance of (d-)StereoTrust. We first discuss the methodology of our experiments in 3.3.1. In Sections 3.3.2 and 3.3.3, we present our results that use Epinions dataset and synthetic dataset respectively.

3.3.1 Methodology

To evaluate performance of proposed models, we compare our models with some other algorithms. We consider two metrics: the *accuracy of prediction* that compares the result of the algorithm with some ground truth; and the *coverage* – fraction of the population that can be evaluated by the trustor, given trustor's limited knowledge.

We compare StereoTrust with the following algorithms.

Feedback Aggregation In this model, If trustor does not know the target agent, it asks other agents across the network and aggregate feedbacks to derive the target agent's trust. Please note that as trustor may not have experience with the probed agent, it cannot identify the dishonest agents, thus may suffer false feedbacks.

EigenTrust EigenTrust [25] uses transitivity of trust and aggregates trust from peers by having them perform a distributed calculation using eigenvector of the trust matrix. Trustor first requests its trusted friends about the target agent's trust. Each opinion of a friend is weighted with the friend's global reputation. To get the wide view of the target agent's information, trustor will continue asking its friends' friends, and so on, until the difference of two derived trusts in two subsequent iterations is smaller than a threshold (convergence). Pre-trust agents (with high global reputation) are used in this model.

Transitive Trust (Web of Trust) This model is based on transitive trust chain. If trustor doesn't know the target agent, it asks its neighbors and its neighbors will ask their neighbors if they do not know the target agent either.

Shortest Path In this variation, agent only chooses the shortest path and ignore the trustworthiness of agents along the path. If multiple shortest trust paths exist, trustor will choose the most reliable one (the agents along the path are the most reliable).

Most Reliable Path In this variation, agent will choose its most reliable neighbor who has the highest trust rating to request for the target agent's trust. If this neighbor does not know the target agent, it continues requesting its own most reliable neighbor. So the most reliable path is found. To avoid infinite requesting, number of hops is limited, in this experiment, is 6. That is to say, if no agent knows the target agent within 6 hops, this model fails to derive the target agent's trust.

Group Feedback Aggregation d-StereoTrust uses opinions reported by the agents who are the members of honest subgroups and agents who are also interested in the corresponding groups as the metrics to measure closeness between the target agent and the subgroups. We compare the accuracy of trust value derived using other agent’s opinions (called group feedback aggregation) with that derived using d-StereoTrust to validate whether such third party information is used by d-StereoTrust judiciously. Different from feedback aggregation we described above, group feedback aggregation only uses the feedbacks provided by the agents that are interested in the corresponding groups.

Dichotomy-only d-Stereotrust divides each group into an honest and dishonest subgroups. To evaluate the impact of the initial grouping in d-Stereotrust, we compare d-Stereotrust with a similar, dichotomy-based algorithm, but without the initial grouping (without the stereotypes). In dichotomy-only, agent a_x classifies all the agents it has previously interacted with into two groups: honest and dishonest (“honest agents” having more successful transactions with a_x). To evaluate agent a_y , similarly to d-Stereotrust, a_x asks honest agents about their trust information to a_y , and using these feedbacks, calculates the distance between a_y and the two groups. If dichotomy-only’s results are similar to d-Stereotrust, the “stereotypes” used in d-Stereotrust (the initial groups) are not needed, as they do not increase the accuracy of the model.

Please note that we only compare StereoTrust model with other trust models using synthetic dataset because we lack essential information of how users in Epinions dataset carry out these existing models. For instance, we cannot detect friend of friend relationship in the dataset so EigenTrust cannot be conducted.

To estimate the accuracy of each algorithm, we compare the value of trust computed by the algorithm for a pair of agents with the ground truth. Then, we aggregate these differences over different pairs using Mean Absolute Error (MAE).

We present the results in two formats. Firstly, to measure overall performance of an algorithm, we show MAE aggregated over the whole population of agents (e.g., Table 3.1). Secondly, to see how the algorithm performs in function of agent’s ground truth, we construct figures presenting the derived trust for a subset of agents (e.g., Figure 3.5.a). To avoid cluttering, we randomly choose 50 target agents. Y-axis represents the trust rating of the agents. X-axis represents the index of the evaluated agent. For clarity, agents are ordered by decreasing ground truth.

Ideally, the ground truth of an agent represents its objective trustworthiness. However, as we are not able to measure it, we have to estimate it using the available data. We will discuss how to derive the ground truth when mapping each dataset. Besides prediction accuracy, we also measure the performance of algorithms using coverage – percentage of agents in the system that can be evaluated by a trustor.

The complete Epinions dataset we crawled contains 5,215 users, 224,500 reviews and 5,049,988 ratings of these reviews. For our experiments, we selected 20 trustors and 150 target agents randomly (we repeated the experiments with different agents and got the similar results). On the result plots (e.g., Figure 3.5.a), error bars are added to show the deviation of predictions by each trustor for the same target agent.

In the experiments using synthetic dataset, we choose one honest agent randomly as trustor to predict behavior of other agents in the system (there are totally 200 agents in the system). Each experiment is repeated 10 times (each running uses different synthetic dataset) and error bars are added to indicate the deviation of each running (e.g., Figure 3.6.b).

3.3.2 Epinions Dataset

Epinions.com is a web site where users can write reviews about the products and services, such as books, cars, music, etc (later on we use the generic term “product”). A review should give the reader detailed information about a specific product. Other users can rate the quality of the review by specifying whether it was *Off Topic*, *Not Helpful*,

Somewhat Helpful, Helpful, Very Helpful or *Most Helpful*. For each review, Epinions.com shows an average score assigned by users.

Epinions.com structures products in tree-like categories. Each category (e.g., books) may include more specific categories (e.g., adventure, non-fiction, etc.). The deeper the level, the more specific category the product belongs to.

Epinions community provides a good scenario to test our proposed model, where users write reviews or rate reviews of products they are interested. This gives the intuitive grouping criteria, that is, users are grouped if they are interested in the same product/category.

3.3.2.1 Mapping of Epinions to StereoTrust Model

To map Epinions.com to StereoTrust model, we treat each user as an agent. Epinions.com categories provide a natural representation of *interested in* relation. A user is *interested in* a (sub)category if she wrote or rated at least one review of a product under this category. Groups are formed according to agents' 'interested in' relations. Consequently, each Epinions.com category corresponds to a group of agents, each of whom is *interested in* (wrote or rated a review for) this category. A transaction between agents a_x and a_y occurs when a_x rates a review written by a_y . To map Epinions.com ratings to StereoTrust binary outcome, we assume that the transaction is successful only if the assigned rate is *Very Helpful* or *Most Helpful*. We set the threshold so high to avoid extreme sparsity of unsuccessful transactions (over 91% review ratings are *Very Helpful* or *Most Helpful*).

We compute the ground truth of an agent as the average rating of the reviews written by this agent. For instance, if an agent wrote 3 reviews, the first review was ranked by two users as 0.75 and 1.0 respectively, while the second and the third received one ranking each (0.75 and 0.5), the ground truth for that user is equal to $(0.75 + 1.0 + 0.75 + 0.5)/4$. The "ground truth" computed with this simple method only approximates the real trustworthiness of an agent, as we do not adjust the scores to counteract, e.g., positive or negative biases of the scoring agents.

3.3.2.2 Results

Figure 3.5.a shows the performance of StereoTrust model. SOF/SOP on the legend indicates that the trust rating is calculated using SOF/SOP approach respectively. From the figure we can see that both the SOF and the SOP approaches fail to provide a good fit to the ground truth. This is because, in Epinions dataset, most ratings given by the agents are positive (*Very Helpful* or *Most Helpful*). So trustors are in a very friendly environment, which makes them difficult to identify the agents who write somewhat low quality reviews by simply using local information.

Figure 3.5.b shows the performance of d-StereoTrust model. We can see that both the SOF and the SOP derived trust ratings are more accurate than feedbacks derived trust rating (group feedback aggregation), which shows that our model outperforms the approach that simply aggregates other agents' feedbacks. SOF approach gives a better fit to the ground truth than SOP approach does. Comparing Figure 3.5.a and 3.5.b, we observe that d-StereoTrust is obviously better than StereoTrust (d-StereoTrust provide better fit to ground truth), so d-StereoTrust improve the accuracy of prediction of the target agent's performance as we expected.

Figure 3.5.c compares d-StereoTrust model with dichotomy-only (StereoTrust is omitted as it is worse than d-StereoTrust in terms of prediction accuracy). Error bars are removed for clarity and only SOF approach, which outperforms SOP approach is showed for each model. From the figure we see that d-StereoTrust model provides more accurate prediction than dichotomy-only does. This proves that considering both interests based group and some global information can predict the target agent's behavior more accurately.

Table 3.1 lists the calculated MAE along with 95% confidence interval. Although plots only show 50 target agents, MAE/95% C.I. is computed using all 20×150 (trustor, target agent) pairs. Confidence interval is computed using the expression $\bar{x} \pm S_E \cdot 1.96$, where $\bar{x}$ is the sample mean, S_E is the standard error for the sample mean, and 1.96 is the 0.975 quantile of the normal distribution.

3.5.a: Comparison of StereoTrust model and 3.5.b: Comparison of d-StereoTrust and ground truth.

3.5.c: Comparison of d-StereoTrust model and dichotomy-only.

Figure 3.5: Experiments results using Epinions.com dataset.

Table 3.1: Mean Absolute Error (with 95% Confidence Interval)

	MAE	95% C.I.
StereoTrust (SOF)	0.1114	(0.1067,0.1161)
StereoTrust (SOP)	0.1177	(0.1136,0.1218)
d-StereoTrust (SOF)	0.0632	(0.0586,0.0678)
d-StereoTrust (SOP)	0.1299	(0.1245,0.1353)
dichotomy-only (SOF)	0.1365	(0.1307,0.1423)
dichotomy-only (SOP)	0.1750	(0.1690,0.1810)
group feedback aggregation	0.1452	(0.1386,0.1518)

3.3.2.3 Discussion

From the results we can see that simply using only “stereotype” (category information) to form group (StereoTrust model) does not predict the target agent’s performance accurately. This is because Epinions.com is a friendly community where users are likely to give high ratings to the reviews written by others, which makes the successful transactions dominate the groups. In such an environment, StereoTrust model will derive

a high rating for the target agent with a high probability. Due to the same reason, dichotomy-only does not work well either. On the contrary, d-StereoTrust improves the prediction accuracy. This proves that using groups and small amount of trust information from other agents helps provide more accurate prediction of the target agent’s performance.

3.3.3 Synthetic Dataset

Epinions.com is a friendly community with few dishonest agents (over 91% review ratings are *Very Helpful* or *Most Helpful*). This hinders a comprehensive study on how StereoTrust performs when a large fraction of agents act dishonestly. So in order to test StereoTrust in a more hostile environment, we generated a synthetic dataset simulating a hostile version of Epinions.com-like community.

3.3.3.1 Synthetic Dataset Generation

In the synthetic dataset, 40% of the population of 200 agents are dishonest. A honest agent provides a high quality review (with real rating = 0.6 or 0.8 or 1.0) or a true feedback with a probability 0.9. A dishonest agent provides a low quality review (real rating = 0.0, 0.2, 0.4) or a false feedback with a probability 0.9. If a true feedback is a rating of λ , the corresponding false feedback is a rating of $1 - \lambda$. Both number of reviews written by an agent and number of ratings of a review are generated by a normal distribution ($\mu = 10$, $\sigma = 4$). Note that the agents who assign ratings to a review are selected randomly from a set of agents who are also interested in the category that this review is about.

We simulate an environment with 12 categories (indexed 1, 2, ..., 12) and 20 products in each category. Honest and dishonest agents are biased towards different categories. A honest agent writes a review for a product under categories 1, 2, 3 or 4 with a probability of 0.7; for a product under categories 9, 10, 11 or 12 with a probability of 0.21 ; and for a product under categories 5, 6, 7 or 8 with a probability of 0.03. A dishonest agent writes a review for a product under categories 5, 6, 7 or 8 with a probability of 0.7; for

a product under categories 9, 10, 11 or 12 with a probability of 0.21; and for a product under categories 1, 2, 3 or 4 with a probability of 0.03 .

We compute the ground truth of an agent as the average rating of the reviews written by this agent. Different from ground truth in Epinions dataset, in synthetic dataset, rating of one review is determined by the design of dataset, so this rating represents the real quality of the review, thus the calculated ground truth better approximates the objective trustworthiness of an agent.

3.3.3.2 Results

Figure 3.6.a shows the performance of StereoTrust model. We can see that the model derived trust rating fits the ground truth in general but not very closely. However, the trend looks better than that in real Epinions dataset.

Figure 3.6.b shows the performance of d-StereoTrust. It is clear that d-StereoTrust provides more accurate prediction than StereoTrust model (more fits to ground truth). This is because, d-StereoTrust forms groups in a finer granularity thus local trust information and third party information are properly used to represent the target agent's trust. Similar to Epinions dataset, feedback derived trust (group feedback aggregation) is less accurate than that derived by SOF/SOP.

Figure 3.6.c compares d-StereoTrust model (using SOF) with dichotomy-only (using SOF) and various existing algorithms (described in section 3.3.1). Error bars are removed for clarity. From the figure we observe that the existing algorithms predict the target agent's trust less accurately than d-StereoTrust does. These existing algorithms show obvious gaps between ground truth and the derived rating for honest target agents part (like EigenTrust and most reliable path transitive trust) or dishonest target agents part (like shortest path transitive trust) or all the target agents (like feedback aggregation).

Table 3.2 summarizes MAE (with 95% confidence interval for all the agents) and coverage of the trust models involved in comparison. For each model, we show MAE for evaluating honest target agents part, dishonest target agents part and all target

3.6.a: Comparison of StereoTrust model and 3.6.b: Comparison of d-StereoTrust and ground truth.

3.6.c: Comparison of all algorithms.

Figure 3.6: Experiments results using synthetic dataset.

agents respectively. For StereoTrust, d-StereoTrust and dichotomy-only, we only show the results using SOF approach, which outperforms SOP approach.

Table 3.2: Mean Absolute Error and Coverage (for synthetic dataset)

Trust models	Honest agents	Dishonest agents	All agents (with 95% C.I.)	Coverage
d-StereoTrust (SOF)	0.1154	0.1099	0.1126 (0.1036,0.1216)	95.5%
EigenTrust	0.1487	0.1002	0.1263 (0.0966,0.1510)	96.4%
Dichotomy-only (SOF)	0.1326	0.1215	0.1288 (0.1202,0.1374)	96.3%
StereoTrust (SOF)	0.1377	0.2641	0.1884 (0.1300,0.2753)	96.9%
Feedback aggregation	0.1450	0.1642	0.1535 (0.1432,0.1678)	99.9%
Transitive trust (shortest path)	0.1547	0.3319	0.2304 (0.1424,0.3384)	99.3%
Transitive trust (most reliable path)	0.1468	0.1678	0.1552 (0.1416,0.1688)	82.1%

3.3.3.3 Discussion

Feedback aggregation and both variations of transitive trust models are significantly worse than d-StereoTrust in terms of prediction accuracy even if feedback aggregation

and transitive trust (shortest path) have the best coverage. Transitive trust model (most reliable path) has the worst coverage. EigenTrust performs almost as good as d-StereoTrust but requires more third party information thus incurring higher communication overhead. Additionally, the assumption of pre-trusted agents is not realistic, which is essential to this model. Dichotomy-only, as a baseline, proves that considering “stereotypes” improves the accuracy prediction of the target agent’s behavior definitely. So to sum up, d-StereoTrust, which has the highest prediction accuracy (MAE is the smallest) at the cost of losing a bit coverage (95.5%) and incurring medium communication overhead (trustor only asks agents that are also interested in corresponding categories) outperforms all the considered models. This also proves that d-StereoTrust is more robust to large portion of malicious agents (up to 40% in the experiments) than other algorithms. Both real and synthetic dataset proves that StereoTrust model does not work as well as d-StereoTrust model. Nevertheless, StereoTrust has its own advantage in terms of communication overhead because it only uses local information to derive trust of the target agent. This is very promising in some scenarios where, by carefully forming groups, honest and dishonest agents are put into groups which seldom overlap, thus group trust can represent individual trust accurately. It can thus be used when information used by traditional trust models, and partly by d-StereoTrust may simply not be available.

3.4 Conclusions

We consider the problem of predicting the trust between a trustor and an unknown agent in a large-scale distributed setting. Traditional approaches to this problem derive unknown agent’s trust essentially by combining trust of third parties to the agent with the trustor’s trust of these third parties; or simply by aggregating third parties’ feedbacks about the unknown agent. In contrast, StereoTrust uses different *kind* of information: that of *semantic* similarity of the unknown agent to other agents that the trustor personally knows. In StereoTrust, a trustor builds stereotypes that aggregate and summarize the experience he had with different kinds of agents. The basis

of the grouping to construct stereotypes is very flexible. For instance, stereotypes can be based on information from the agents' personal profiles, or the class of transactions they make. Facing a possible transaction with an unknown agent, the trustor builds its trust by cumulating the experience from the stereotypes to which the unknown agent conforms.

The stereotypes are based entirely on the local perspective and local information of the trustor, and, therefore, are naturally suited for large-scale systems; personalized for each trustor; and less susceptible to false or unsuitable information from third parties. The rationale for the StereoTrust approach is to determine an alternative and complementary mechanism (than existing techniques) to compute trust even in absence of (global) information that is likely to be unavailable under some circumstances, and instead using some other class of information (stereotypes) which can be established by local interactions.

When some of third parties' opinions about an agent are available, we propose an enhancement (d-StereoTrust), which creates a "good" and a "bad" subgroup inside each stereotype. The trustor assigns each one of his previous transaction partners to one of these groups based on his personal experience with the partner (e.g., the ratio of the failed transactions). Then, the trustor uses the aggregated third parties' opinions about the unknown agent to determine how similar is the agent to the "good" and "bad" subgroup. Third parties' opinions are a small subset of information used by traditional mechanisms (such as feedback aggregation or Eigentrust-type algorithms). However, according to our experiments, by combining stereotypes with these partial historical information, d-StereoTrust predicts the agent's behavior more accurately than Eigentrust and feedback aggregation.

StereoTrust, however, has its own shortcomings. For instance, it cannot accurately tell which stereotypes are more important than others; the transaction amounts based weight determination for stereotypes combination is straightforward, intuitive and heuristic, etc. In order to solve these problems, we propose an enhancement by leveraging on machine learning algorithms, which will be elaborated in the next chapter.

Publication(s)

- *StereoTrust: A group based personalized trust model*

Liu Xin, Anwitaman Datta, Krzysztof Rzadca, Ee-Peng Lim

In Proceedings of the 18th ACM conference on Information and knowledge management (CIKM), Hong Kong, China, 2009.

Chapter 4

A machine learning framework for identifying relevant features to determine trust

In the previous chapter, we introduced the StereoTrust model, and showed how it works without the target agent’s complete historical information. However, StereoTrust simply aggregates stereotypes by assigning intuitive weights to derive trust of the target agent, thus is not able to accurately tell which stereotypes are more important than other ones¹. So in order to handle such shortcomings in StereoTrust, we next propose MetaTrust [45, 46], a machine learning framework for identifying relevant features to determine trust. Specifically, a trustor uses its own previous transactions (with other agents) to build a knowledge base, and utilize this to assess the trustworthiness of a transaction (instead of the target agent) based on associated features, which are capable of distinguishing successful transactions from unsuccessful ones. These features are harnessed using appropriate machine learning techniques (e.g., linear discriminant analysis, decision tree, etc.) to extract relationships between the potential transaction and previous transactions. Different from StereoTrust, by applying machine learning algorithms, MetaTrust is capable of distinguishing which feature (i.e., stereotype) is more discriminating thus making the prediction more accurate.

In case a trustor does not have sufficient past transactions, we propose to construct an overlay network to share local knowledge among trustworthy agents, which helps to

---

¹The weights for stereotypes are relatively intuitive and straightforward thus may not be able to in all cases reflect the real importance of each stereotype

bootstrap inexperienced users. Such extension of the MetaTrust model thus allows a seamless incorporation of existing techniques based on aggregation of historical data as well as social (web-of-trust) networks.

We introduces the basic idea of MetaTrust in Section 4.1. The machine learning based trust framework is introduced in Section 4.2. In Section 4.3, we demonstrate how MetaTrust works using specific machine learning algorithms, namely linear discriminant analysis (LDA) and decision tree (DT) as case studies, which are presented in subsections 4.3.1 and 4.3.2 respectively. In order to determine how confident the trust prediction provided by our approach is, we study the confidence of the LDA and DT based algorithms in Section 4.4. We then discuss the issue of sharing local knowledge among the agents in Section 4.5.

4.1 Introduction

As mentioned in the introduction chapter, our main goal (in this part of thesis) is to estimate trust without the target agent’s complete historical information. StereoTrust demonstrates our first step to achieve this goal. With traditional beta probability function based method, StereoTrust is able to derive and update an agent’s stereotypes effectively. However, given a set of features, StereoTrust fails to discern the most useful ones for stereotypes formation and efficiently combine the same. This thus leaves big room for accuracy improvement and gives the objectives of MetaTrust: some stereotypes that are based on more discriminating features should have greater impact on the final prediction than the ones that are based on less discriminating features, and doing so needs a more sophisticated mechanism, one possibility being the use of machine learning techniques, as is carried out in this work. Please note that slightly different from StereoTrust, MetaTrust focuses on estimating trustworthiness of the potential transaction, instead of the the target agent (who will participate in the transaction).

MetaTrust is a machine learning techniques based trust framework, so various machine learning algorithms can be integrated. In this chapter, we use two common but

effective algorithms: linear discriminant analysis (LDA) [47, 93] and decision tree (DT) [48, 42] as the case studies for presentation. LDA is a well known method for dimensionality reduction and classification. It takes as input a set of events belonging to k (≥ 2) different classes and characterized by various features, and finds a combination of the features (a classifier) that separates these k classes of events. As an example, linear discriminant analysis was used to differentiate subspecies of beetles based on measurements of their physical characteristics. Decision tree is a widely used and practical method for classification and prediction by providing a classifier in the form of a tree structure. Its popularity is due to the ability of generating rules which can be easily expressed visually and in human language. Although LDA and DT are simple, as seen in our real auction data driven experiments (presented in Chapter 6.4), they are both effective and efficient.

In MetaTrust, similar to StereoTrust, an agent’s local knowledge (i.e., the past interactions) are described by a set of features². We assume two classes: successful and unsuccessful interactions. For LDA, features must be quantitative, but can take values on an unbound domain. The agent who needs to evaluate the potential interaction divides his historical interactions into two groups: successful and unsuccessful. He then performs a LDA on this two groups to obtain a linear classifier that allows him to estimate whether the potential interaction is likely to get classified in the successful group or not. For DT, the features can be quantitative or qualitative. A decision tree is firstly constructed from the training data. Then the algorithm classifies an example data by starting from the root of the tree and moving (down) until a leaf node, which is actually the classification of the example data. Since different machine learning algorithms may produce different results in the same scenario (due to their suitability), we also investigate confidence of each algorithm recommendation to help trustor make the wiser decision.

In case trustor does not have sufficient local knowledge, we propose to construct a local knowledge sharing overlay network (LKSON), which allows local information

---

²Note that all the interactions have the same feature set.

(i.e., intermediate results of machine learning algorithms) to be exchanged between trustworthy agents. Such a mechanism helps inexperienced agents to bootstrap in the system.

The main contribution of this chapter is MetaTrust, a machine learning framework for identifying relevant features (for stereotypes formation) to determine trust. This work follows StereoTrust (i.e., assessing trust without the target agent’s past behavior) but greatly improves it in three aspects: (1) stereotypes are efficiently combined using machine learning algorithms such that the more discriminating stereotypes have greater impact than other ones; (2) confidence of the algorithm recommendations is studied to help the trustor make decision wisely; (3) MetaTrust allows efficient information sharing to bootstrap inexperienced agents, making it applicable in a broader range (compared to original StereoTrust).

4.2 System model

4.2.1 Notation

We denote by $\mathcal{A}$ the set of all agents (participants) in the system. There are two types of agents, *customers*, which request service from other agents and *providers*, which provide service. An agent can be both a customer and a service provider but in a specific transaction, he plays only one role. Θ_{a_x, a_y} denotes the transaction between provider agent a_y and customer agent a_x . Note that without loss of generality, each Θ represents a unique transaction. Each transaction Θ is described by a set of features, denoted by $F_\Theta = \{f_\Theta^1, f_\Theta^2, \dots, f_\Theta^d\}$. Each agent a_x in the system maintains a transaction database $Tr_{a_x} = \{\Theta_{a_x, a_1}, \Theta_{a_x, a_2}, \dots\}$, which records historical transactions of this agent with other agents.

4.2.2 Feature collection

In StereoTrust (Chapter 3), we do not discuss how features are collected for stereotypes formation. Here we supplement such discussion for both MetaTrust and StereoTrust within a specific application scenario.

Features can be collected from profile of the transaction partner, or the context of the transaction depending on specific applications. Let us take online auction as an example to demonstrate how features are collected. Buyer Bob is evaluating a potential transaction of purchasing a camera from the seller Sally. The features associated with this transaction can be collected from three aspects:

- (1) about Sally herself, e.g., age, gender, is she from the same country as bob? physical distance to bob, etc.
- (2) about Sally as a user in the system, e.g., age in the system, numbers of successful and unsuccessful transactions, does Sally has provided a phone number? how complete is her profile? number of items Sally already sold, average delivering time, average time between end of auction and user comment, number of friends Sally has, etc.
- (3) the cameras Sally sells, e.g., average item price (items in the same category), number of the same items in stock, number of comments on these cameras, number of different buyers that already placed bid on it, average age in the system of buyers that already placed bid, etc.

It is worth mentioning that Sally can cheat and set fake information about herself, but faking information from the *2nd* and *3rd* aforementioned categories is harder since this information is maintained by the service exchange platform.

We argue that compared to existing reputation based trust models, the feature collection process is more robust against malicious agents: for a traditional reputation based trust model, when a trustor requests opinions about the target agent, malicious entities can easily counterfeit the feedbacks to promote or bad-mouth the target agent and the trustor needs much efforts to filter out these false feedbacks, but for StereoTrust and MetaTrust, since attackers are unaware of the trustor’s local knowledge, it is difficult to fake features associated with the potential transaction to mislead the trustor. StereoTrust and MetaTrust are thus more resilient to the attacks that are common to current reputation based trust systems.

4.2.3 Architecture

Fig. 4.1 depicts the architecture of MetaTrust. The framework consists of three components: storage component (SC), trust calculation engine (TCE) and knowledge collector (KC).

Figure 4.1: The architecture.

Storage component is responsible for managing local knowledge, including agent's past transactions as well as third party knowledge collected from other agents. After each transaction, agent records associated information such as characteristics of the transaction partner, outcome of the transaction and other related features (done by feature extraction engine), depending on the types of the applications. For instance, in online auction site, such features may be price or category of the item, or number of items already sold by the seller, etc. (see Table 4.1 as an example using Allegro [64] dataset). These past transactions are used as the knowledge base to perform machine learning algorithms.

MetaTrust is generic in two senses: (1) like StereoTrust, MetaTrust can be applied for different applications and correspondingly diverse features can be identified and utilized; and (2) diverse machine learning approaches may be plugged in the framework. Trust calculation engine is responsible for applying machine learning algorithms to predict trustworthiness of the potential transaction using local knowledge or collected knowledge

Table 4.1: Structure of local knowledge repository

Tran. ID	Tran. part- ner	outcome	Price of the item	# of items already sold	...
Θ_1	a_1	1	6.99	891	...
Θ_2	a_2	0	9.99	337	...
Θ_3	a_3	0	12.99	120	...
.....					

as the training set. Depending on application scenario, the TCE is able to choose the appropriate machine learning algorithm to conduct trust calculation. For instance, if features are quantitative, discriminant analysis will be used, otherwise, if features are qualitative, decision tree is more suitable. Actually, subject to time/computation restriction, TCE will apply as many suitable algorithms as possible. In case the result is not consistent across all algorithms, TCE returns the result that is calculated by the algorithm which has the highest confidence. Measurement of such confidence is algorithm dependent and we will discuss this issue in Section 4.4.

With sufficient local knowledge and suitable machine learning algorithms, trustor is able to reliably predict trustworthiness of a potential transaction. However, when local knowledge is insufficient, machine learning algorithms will perform poorly. To address this issue we propose to construct a local knowledge sharing overlay network (LKSON) where agents are able to share their local information (done by knowledge collector (KC)). Different from traditional trust mechanisms where feedbacks of the specific agent are shared, in MetaTrust, agents only exchange intermediate machine learning algorithm result (see Section 4.5). Such a strategy has several advantages: (1) the shared information is just intermediate result of an algorithm so it is not easy to dig out agent's privacy, e.g., identification; (2) since information provider does not know whom the trustor is evaluating as well as trustor's local knowledge, it is difficult to send fake information to promote or bad-mouth the specific agent; (3) a lot of computation duplication is avoided. We will present the local knowledge sharing overlay network in detail in Section 4.5. We next present details of our machine learning based trust assessment algorithms.

4.3 Machine learning algorithms for trust assessment

A machine learning algorithm for trust assessment takes as input a set of trustor's past transactions with other agents and the feature vector associated with each transaction. The output of the algorithm is a prediction indicating whether the potential transaction is reliable or risky.

We next use two common but effective machine learning algorithms, i.e., linear discriminant analysis and decision tree as the case studies to demonstrate how MetaTrust works. Algorithm recommendation confidence is discussed in Section 4.4.

4.3.1 Case study: Using linear discriminant analysis

Consider a scenario where a customer a_x encounters a potential service provider a_y and a_x has no prior experience with a_y . We assume that a_x can obtain features of this potential transaction Θ_{a_x, a_y} (e.g., from a_y 's personal profiles and from the transaction offer it received). The set of such features of Θ_{a_x, a_y} is denoted by $F_{\Theta_{a_x, a_y}} = \{f_{\Theta_{a_x, a_y}}^1, f_{\Theta_{a_x, a_y}}^2, \dots, f_{\Theta_{a_x, a_y}}^d\}$. (to simplify the notation, we use f^i instead of $f_{\Theta_{a_x, a_y}}^i$ when the context is clear). So the potential transaction is represented by vector $p = (f^1 \ f^2 \ f^3 \ \dots \ f^d)$.

We assume that a_x has recorded n historical transactions. To estimate reliability of the potential transaction, a_x classifies its historical transactions into two disjoint groups, the *successful transaction group* G_s and the *unsuccessful transaction group* G_u according to the outcomes of these transactions. The two transaction groups are represented as³:

$$\mathbf{G}_s = \begin{pmatrix} f_{\Theta_{a_x, a_1}}^1(s) & f_{\Theta_{a_x, a_1}}^2(s) & \dots & f_{\Theta_{a_x, a_1}}^d(s) \\ \vdots & \vdots & \vdots & \vdots \\ f_{\Theta_{a_x, a_{n_s}}}^1(s) & f_{\Theta_{a_x, a_{n_s}}}^2(s) & \dots & f_{\Theta_{a_x, a_{n_s}}}^d(s) \end{pmatrix} \quad (\text{Eq. 4.1})$$

---

³We assume an agent is aware of context of the transaction. For instance, when evaluating an unknown book seller, the buyer only considers past transactions regarding purchasing books, not that regarding purchasing other irrelevant items like digital cameras.

$$\mathbf{G}_u = \begin{pmatrix} f_{\Theta_{a_x, a_1}}^1(u) & f_{\Theta_{a_x, a_1}}^2(u) & \dots & f_{\Theta_{a_x, a_1}}^d(u) \\ \vdots & \vdots & \vdots & \vdots \\ f_{\Theta_{a_x, a_{n_u}}}^1(u) & f_{\Theta_{a_x, a_{n_u}}}^2(u) & \dots & f_{\Theta_{a_x, a_{n_u}}}^d(u) \end{pmatrix} \quad (\text{Eq. 4.2})$$

n_s and n_u are the sizes of successful transaction group and unsuccessful transaction group respectively. Please note that $n = n_s + n_u$.

a_x performs linear discriminant analysis to classify the potential transaction as belonging to successful or unsuccessful transaction group to decide whether or not to transact with the corresponding service provider for the specific transaction. The mathematical operations are the following. Let h_x be a $x \times 1$ (column) vector of ones.

a_x first calculates centroid of each group. That is, for each feature, we calculate average value of this feature across all transactions in the group.

$$c_s = \frac{1}{n_s} \cdot h_{n_s}^T G_s \text{ and } c_u = \frac{1}{n_u} \cdot h_{n_u}^T G_u, \quad (\text{Eq. 4.3})$$

Similarly, the global centroid is calculated by averaging each feature across all past transactions:

$$c = \frac{1}{n} \cdot h_n^T \begin{bmatrix} G_s \\ G_u \end{bmatrix} \quad (\text{Eq. 4.4})$$

In LDA, the internal variance (within-class scatter matrix) and external variance (between-class scatter matrix) are used to indicate the degree of class separability, i.e., to what extent are the successful transactions distinguished from the unsuccessful transactions. The internal variance, which is the expected covariance of each group is given by Eq. 4.5 and Eq. 4.6:

$$S_w^s = \frac{1}{n_s} (G_s - h_{n_s} c_s)^T (G_s - h_{n_s} c_s), \quad (\text{Eq. 4.5})$$

$$S_w^u = \frac{1}{n_u} (G_u - h_{n_u} c_u)^T (G_u - h_{n_u} c_u). \quad (\text{Eq. 4.6})$$

h_{n_s} and h_{n_u} are used to center the values around the centroids c_s and c_u . So the overall within-class scatter matrix is calculated as the weighted sum of each group's internal variance, where the weight is fraction of transactions regarding the corresponding group:

$$S_w = \frac{1}{n}(n_s S_w^s + n_u S_w^u). \quad (\text{Eq. 4.7})$$

Then a_x calculates external variance, which is actually the covariance of the two groups, each of which is represented by its mean vector (Eq. 4.8).

$$S_b = \frac{1}{n}(n_s(c_s - c)^T(c_s - c) + n_u(c_u - c)^T(c_u - c)), \quad (\text{Eq. 4.8})$$

The mixture scatter matrix, i.e., the total variance of the system is obtained by summing up internal variance S_w and external variance S_b (Eq. 4.9).

$$S_m = S_w + S_b. \quad (\text{Eq. 4.9})$$

LDA aims to find a projection direction (a transformation) v that maximizes the inter class variance and minimizes the intra class variance. Formally, the criterion function (see Eq. 4.10) is to be maximized.

$$J(v) = \frac{v^T S_b v}{v^T S_w v} \quad (\text{Eq. 4.10})$$

The projection direction v is found as the eigenvector associated with the largest eigenvalue of $S_w^{-1} S_b$. We then transform the two groups of transactions using v . Similarly, the potential transaction $p = (f^1 \ f^2 \ f^3 \ \dots \ f^d)$ is also transformed and classified by measuring the distances between transformed potential transaction and the two groups (i.e., centroid). Eq. 4.11 and Eq. 4.12 show how to calculate such distances:

$$D_s = v^T p - v^T c_s \quad (\text{Eq. 4.11})$$

$$D_u = v^T p - v^T c_u \quad (\text{Eq. 4.12})$$

If $D_u > D_s$, then transaction p is predicted as successful, otherwise it is predicted as unsuccessful.

Note that we try to collect as many features of a transaction as possible, and the algorithm filters out the not-so-relevant variables for us. That is to say, the feature which is more capable of distinguishing successful transactions from unsuccessful ones will have more impact on the final classification result. This is exactly the advantage of MetaTrust compared to StereoTrust.

4.3.2 Case study: Using decision tree

The basic idea of decision tree is to classify objects by sorting them from root of the tree to some leaf nodes, which provide classifications of the objects. Each node (except the leaf nodes) in the tree represents some feature of the object, and the value of such feature is the criteria by which the current set of objects are put in the final classifications or further split into smaller classes that will be classified according to other selected features (i.e., child nodes of the current node). This process continues until all objects are classified.

When encountering a potential transaction from an unknown agent, trustor a_x has n past transactions with other agents $Tr_{a_x} = \{\Theta_1, \Theta_2, \dots, \Theta_n\}$. Each past transaction Θ is described by the same set of features $F_\Theta = \{f_\Theta^1, f_\Theta^2, \dots, f_\Theta^d\}$. Such knowledge is used to construct the decision tree to generate a classifier (rule) to predict whether a potential transaction is risky or not⁴.

From a given past transaction set, it is possible to construct several different decision trees, depending on the order in which transaction features are tested. Since building an optimal decision tree (e.g., optimal with respect to its size) is NP hard [94] and therefore, several heuristics have been proposed to build efficient learning trees (e.g., ID3 [48], C4.5 [95], etc.). Without loss of generality, we hereafter present one of these heuristics, namely ID3 [48]. ID3 is a well known algorithm that relies on information

---

⁴Recall that we assume binary result of a transaction.

gain [96] to select the classifying features at each node of the tree. Information gain is measured by entropy.

Given the binary categorization (i.e., successful and unsuccessful) and past transactions, we denote the proportion of successful and unsuccessful transactions by p_s and p_u respectively. Then the entropy of all past transactions is:

$$Entropy(Tr_{a_x}) = -p_s \log_2(p_s) - p_u \log_2(p_u) \quad (\text{Eq. 4.13})$$

Entropy is used to characterize (im)purity of a collection of examples. From Eq. 4.13 we can see entropy will have the minimum value of 0 when all past transactions belong to one class (i.e., successful or unsuccessful) and the maximum value of $\log_2 2 = 1$ when past transactions are evenly distributed across the two classes. Using entropy, we now calculate information gain of every feature (with respect to a set of past transactions Tr) to determine the best feature to choose for a node in the decision tree. For each feature f_i , we assume it has a set of values (e.g., discrete variable) or intervals (e.g., continuous variable), which is denoted by $V(f_i)$. For each $v \in V(f_i)$, we denote the set of past transactions that are associated with v for feature f_i by Tr_v . The information gain of feature f_i is thus calculated by:

$$IGain(Tr, f_i) = Entropy(Tr) - \sum_{v \in V(f_i)} \frac{|Tr_v|}{|Tr|} Entropy(Tr_v) \quad (\text{Eq. 4.14})$$

The information gain of a feature measures expected reduction in entropy by considering this feature. Clearly, the higher the information gain, the lower the corresponding entropy becomes and thus the better the classification of past transactions is achieved (by considering the corresponding feature).

Using information gain, a decision tree is iteratively constructed by maximizing information gain at each step. We next describe how decision tree is constructed (Algorithm 1) by ID3 algorithm.

The ID3 algorithm input is trustor a_x 's past transactions Tr_{a_x} and the feature vector F that describe each transaction. If all transactions are successful or unsuccessful, the

---

Algorithm 1 Decision tree construction (code for trustor a_x).

```

1: Trust_ID3( $Tr_{a_x}, F$ )
2: if All past transactions are successful then
3:   Return single node tree  $Root$ , with label = +.
4: end if
5: if All past transactions are unsuccessful then
6:   Return single node tree  $Root$ , with label = -.
7: end if
8: if  $F = \emptyset$  then
9:   Return single node tree  $Root$ , with label = outcome of the transactions that
      dominate all past transactions.
10: end if
11: Calculate information gain of all features and choose one feature (say,  $f$ ), which has
      the highest information gain as the  $Root$  node.
12: for each value  $v$  of  $f$  do
13:   Add a new branch below  $Root$ .
14:   Let  $Tr_{a_x, f, v}$  be the set of past transactions that have value  $v$  for feature  $f$ .
15:   if  $Tr_{a_x, f, v} = \emptyset$  then
16:     Below this branch add a leaf node with label = outcome of the transactions
        that dominate all past transactions.
17:   else
18:     Below this branch add a sub-tree:
19:     Trust_ID3( $Tr_{a_x, f, v}, F - \{f\}$ ).
20:   end if
21: end for
22: Return  $Root$ .

```

---

algorithm simply returns a single node (i.e., root) tree, labeled as '+' or '-' respectively (Line 2-7). If feature vector is empty, the algorithm returns a single node tree, labeled as '+' if most of past transactions are successful or '-' otherwise (Line 8-10). Then the algorithm calculates information gain of each feature using Eq. 4.14. The feature (say, f) with the highest information is selected as the root of the tree. Next, new branches are added to the root node according to possible values of f . For each possible value v , we denote the set of past transactions that have v for f by $Tr_{a_x, f, v}$. If $Tr_{a_x, f, v} = \emptyset$, the algorithm adds a leaf node to the corresponding branch with label = '+' or '-' if most of past transactions are successful or unsuccessful respectively. Otherwise, the algorithm recursively executes itself with $Tr_{a_x, f, v}$ and $F - \{f\}$ as the input parameters (Line 12-21). The algorithm terminates either when all the features have been tested or the tree perfectly classifies the past transactions.

When encountering a potential transaction, a_x tries to classify it as a risky transaction or not by comparing the corresponding features from root of the tree down to a certain leaf node. Note that for simplicity, we do not discuss other issues in decision tree such as avoiding overfitting the data, etc.

4.4 Algorithm recommendation confidence

Given a features vector, a set of past transactions and appropriate machine learning algorithm, MetaTrust is able to return a recommendation of the potential transaction: be it risky or not. Confidence of such recommendation relies on characteristics of input (e.g., discriminating power of the features, volume of past transactions) and performance of the specific machine learning algorithms in use⁵. So by performing the same algorithm, trustors with different local knowledge and/or feature vectors may have different confidence on the recommendations. We denote $\sigma \in [0, 1]$ as the algorithm recommendation confidence, where 0 represents the algorithm recommendation is not confident at all (i.e., the recommendation is equally random) and 1 represents the algorithm recommendation is completely confident. Algorithm input is denoted by $\langle F, Tr \rangle$, where $F = \{f_1, f_2, \dots\}$ represents the feature vector and $Tr = Tr_s \cup Tr_u$ represents the set of past transactions: $Tr_s = \{\theta_1^s, \theta_2^s, \dots\}$ and $Tr_u = \{\theta_1^u, \theta_2^u, \dots\}$ are the sets of successful transactions and unsuccessful transactions respectively. We next present that given different inputs, how to measure confidence of the algorithm recommendations still using linear discriminant analysis and decision tree as the demonstration examples.

4.4.1 Measuring recommendation confidence for LDA based algorithm

According to theory of linear discriminant analysis based algorithm (refer to Section 4.3.1), each transaction is described by the input feature vector F (i.e., each transaction is represented as a point in $|F|$ -dimensional space.). So confidence of the algorithm

---

⁵Given the same input, different machine learning algorithms may generate different recommendations (perhaps due to their suitability in different application scenarios).

recommendation greatly depends on discriminating power of the input features. If all the features are discriminating enough, LDA is able to separate the successful and unsuccessful transactions clearly and minimize the variance of each transaction group. Otherwise, the two transaction groups Tr_s and Tr_u may overlap, thus making classification of the potential transaction less confident. In other words, algorithm input determines the LDA projection direction. After having the project direction, transaction group centroids are transformed to lower dimension space. The potential transaction (after transformed) is classified as risky or not by measuring the distances between it and the two transaction group centroids. We denote the distances between the potential transaction and successful/unsuccessful transaction group by D_s/D_u . Note that each set of distances correspond to one specific input: $\langle D_s, D_u \rangle = \text{LDA}(\langle F, Tr \rangle)$, where $\text{LDA}()$ performs linear discriminant analysis to generate distances using the given input. The confidence of the algorithm recommendation is then calculated as:

$$\sigma_{LDA} = \frac{|D_s - D_u|}{D_s + D_u} \quad (\text{Eq. 4.15})$$

It is clear from Eq. 4.15 that when $D_s = D_u$, $\sigma_{LDA} = 0$, i.e., the potential transaction is classified as risky or not with the same probability. When D_s or D_u is 0, $\sigma_{LDA} = 1$, i.e., the potential transaction is recommended to be safe or not confidently. With σ_{LDA} , trustor is able to measure how confident the algorithm outcome is and personally decides whether or not to trust the algorithm recommendation.

4.4.2 Measuring recommendation confidence for DT based algorithm

Since decision tree is constructed according to information gain of each feature (refer to Section 4.3.2), it is natural to measure algorithm recommendation confidence using feature's information gain. According to information theory, information gain of a feature measures expected reduction in entropy by considering this feature. If a feature is discriminating enough, the corresponding information gain is high (maximum value is

$\log_2 2 = 1$), i.e., by appropriately choosing values of this feature, the associated successful transactions and unsuccessful transactions are separated clearly, thus the potential transaction can be accurately classified by this feature. Otherwise, this feature may not help classify the potential transaction accurately. So with different input data (i.e., feature vector and set of past transactions), different information gains for a specific feature may be generated and hence different confidence on the classification results may be obtained.

For each feature f_i (with respect to a set of transactions Tr_i), we calculate the corresponding entropy $Entropy(Tr_i)$ and information gain $IGain(Tr_i, f_i)$ using Eq. 4.13 and Eq. 4.14 respectively. We then calculate the extent of entropy reduction by feature f_i :

$$r_i = \frac{IGain(Tr_i, f_i)}{Entropy(Tr_i)} \quad (\text{Eq. 4.16})$$

r_i can be interpreted as the accuracy of classification of the potential transaction by feature f_i (i.e., at the node corresponding f_i in the decision tree.). We notice from Eq. 4.16 that if f_i is extremely discriminating, its information gain is $Entropy(Tr_i)$ (i.e., the second part of Eq. 4.14 is 0) and thus r_i is 1, which means the potential transaction classification by f_i is completely accurate. If feature f_i is incapable of discriminating, its information gain is 0 and thus r_i is 0, which means classification of the potential transaction is inaccurate at all (i.e., the classification is equally random.).

After having accuracy of classification of each feature, we calculate algorithm recommendation confidence as:

$$\sigma_{DT} = \prod_{i=1} r_i \quad (\text{Eq. 4.17})$$

So, if all the features are quite discriminating, the algorithm recommendation confidence is approaching 1. Otherwise, any less discriminating feature shall greatly impact the final confidence. This also follows the decision rule of the decision tree based algorithm.

4.5 Local Knowledge Sharing Mechanism

Like StereoTrust, current MetaTrust works under the assumption that the trustor has sufficient local knowledge such that machine learning algorithms can be performed. However, in some cases, agents may have no or sparse local knowledge (e.g., the bootstrapping agents recently join the system or do not interact with others frequently). To make MetaTrust adaptive to such a scenario, we propose to construct a overlay network dedicated to exchange/share agents' local knowledge to help inexperienced agents estimate trustworthiness of a potential transaction.

4.5.1 Local Knowledge Sharing Overlay Network (LKSON)

To evaluate trustworthiness of a potential transaction, inexperienced agents have to request other agents' knowledge. A local knowledge sharing overlay network (LKSON), which is a virtual network on top of current network infrastructure (e.g., P2P network) is thus constructed. Fig. 4.2 depicts a LKSON, which is represented by a directed graph. We give the definition of LKSON:

Definition 4.2 (Local Knowledge Sharing Overlay Network) *A local knowledge sharing overlay network (LKSON) is a Weighted Directed Network (WDN), denoted by $LKSON = (V, E, T)$:*

$V = \{a_1, a_2, a_3, \dots\}$ *is the set of agents in LKSON.*

$E = \{e_{i,j} | a_i \in V \wedge a_j \in V \wedge a_i \neq a_j\}$ *is the set of directed edges representing the trust relationship in terms of providing correct local knowledge between knowledge requester a_i and requestee a_j .*

$T : E \rightarrow [0, 1]$, *trust score $t_{i,j}$ is assigned to edge $e_{i,j}$ and expresses from viewpoint of a_i , how trustworthy a_j is in terms of providing local knowledge.*

We set that the trust score falls into the range of $[0, 1]$, where 1 represents completely trustworthy and 0 represents completely untrustworthy. Trust score is determined according to requestee's past performance (i.e., quality of provided local knowledge). After

each interaction, knowledge requester a_i evaluates usefulness of the information provided by the requestee a_j and accordingly updates trust score of a_j . Please note that since agent may have very personalized perspective on the system, even if requestee a_j provides correct local knowledge, it may not be useful to the requester. In this situation, a_j 's trust score will be lowered although he is honest. We will discuss details of trust score update in Section 4.5.2.

Figure 4.2: A local knowledge sharing overlay network.

Notice that LKSON is constructed to ensure reliable local knowledge sharing to help inexperienced agents estimate trustworthiness of a potential transaction, but is not used to discover trustworthy transaction partner (i.e., LKSON is not a trust overlay network like that in [28]) because it has been reported in related literature [26], [97] that the agents who provide high quality service may not report genuine feedbacks and vice versa. In the case that the agents who provide useful information also act honestly in a transaction, LKSON can be used to help promote successful transactions (i.e., selecting reliable agents who have high trust scores as the service providers). However, the goal of this work is to design mechanisms to estimate trustworthiness of a potential transaction, which is conducted by an unknown agent (i.e., no historical information is available), so

discussion on relying LKSON to estimate agent’s trustworthiness like traditional trust mechanisms (e.g., feedbacks aggregation [19], [33], [80], web of trust [17] [36], etc.) is out of the scope of this chapter.

In order to collect useful information, inexperienced agents only request the agents who are trustworthy (i.e., trust score is high) in terms of providing useful local knowledge. So in MetaTrust, each agent a' maintains a *Trusted Knowledge Provider* list $PR_{a'}$ which stores the agents that this agent has trust relationship in terms of providing useful local knowledge. Initially, when requestees’ past performance is not available, an inexperienced agent requests information by exploring the social network. For instance, it may request its “familiar” agents (e.g., friends or colleagues in the real world, etc.); or the agent may request “special” agent/entity (if any) in the network (e.g., super peer in hybrid peer-to-peer network, or advisor in Epinions [98], or authorized third parties, e.g., “official” source in public key infrastructure, etc.). In the case that no “familiar” or “special” agents are there (i.e., no useful information at all), the inexperienced agent will choose knowledge provider randomly to initiate its *Trusted Knowledge Provider* list.

LKSON is designed to collect input information for MetaTrust in case the trustor has no sufficient local knowledge. During the early stage of an inexperienced agent, LKSON helps bootstrap the agent by aggregating knowledge from other (experienced) agents. After each transaction, the agent updates its trusts in the information providers in terms of usefulness of the provided information (see Section 4.5.2). This indeed adds extra overheads to the system. However, we argue that such overheads are tolerable and LKSON is feasible in real settings: (1) an agent only relies on LKSON for aggregating information when he has insufficient local knowledge. When enough local knowledge has accumulated, this agent just stops requesting other agent’s information (but will provide information when requested⁶); (2) Different from other similar mechanisms, e.g., PeerTrust [26] that considers multiple trust factors, or PowerTrust [28] that uses two

---

⁶We do not discuss how to address the issues of free riders, whitewashers, sybil attack, etc, which are universally existent in any similar systems. Please refer to [99] for a survey on attack and defense techniques for reputation systems.

DHTs to construct a complex trust overlay network, or EigenTrust [25] that maintains a matrix of global reputation scores for all peers in the system (and involves complex reputation calculation), LKSON is a lightweight mechanism that only maintains a *Trusted Knowledge Provider* list for each inexperienced agent (and trust score calculation is straightforward).

We next present details of LKSON design using LDA as the demonstration example. Some issues like what kind of information is shared, how to combine knowledge of multiple sources and how to update relationships among agents in LKSON, etc. will be discussed.

4.5.1.1 What is Shared?

To help an inexperienced agent detect a potentially risky transaction, two types of local knowledge, which are represented as a tuple $\langle v, (c_s, c_u) \rangle$ are shared among the agents⁷:

- (i) the projection direction v that maximizes the inter class variance and minimizes the intra class variance (see Section 4.3.1).
- (ii) centroid of the successful group c_s and unsuccessful group c_u .

The main reasons that these two types of knowledge are shared are: (1) the knowledge providers have no motivation to send fake information because they are unaware of requester’s potential transaction partner thus are unable to defame/promote the transaction partner or mislead the requester effectively⁸. (2) the projection direction v is shared, which means requester does not need to compute it so a lot of computation is saved. The process of deriving projection direction is the most computation-intensive part of LDA. (3) the shared knowledge is just projection direction and centroid of successful and unsuccessful groups so it is not easy to dig out agent’s privacy through

---

⁷We assume that all the agents have the same features such that the projection direction v can be shared and combined meaningfully.

⁸This reason is sometimes not true (e.g., the knowledge provider may learn requester’s transaction partner from other ways, or the provider simply sends incorrect information for fun, etc.). We address this issue by updating *Trusted Knowledge Provider* list, which will be described in Section 4.5.2.

shared information. This is important to knowledge providers and their past transaction partners. (4) The shared knowledge is of small size so the communication overheads incurred are marginal compared to the cost of sending the whole local knowledge base.

4.5.1.2 Combining Third Party Knowledge

When an agent a_i (with no local knowledge) encounters a potential transaction $p = (f^1 f^2 f^3 \dots f^d)$, where f represents a feature. a_i requests local knowledge of other agents that are recorded in a_i 's *Trusted Knowledge Provider* list $PROVIDER(a_i)$ to predict trustworthiness of p . We denote the set of knowledge providers who response and provide the requested information by $KP_i = \{a_1^i, a_2^i, a_3^i, \dots, a_{n_i}^i\}$. The corresponding trust scores of these knowledge providers are denoted by $TS = \{t_1^i, t_2^i, t_3^i, \dots, t_{n_i}^i\}$, where t_j^i represents a_i 's trust in the j th knowledge provider in terms of sharing local knowledge ($\langle v_j, (c_{s,j}, c_{u,j}) \rangle$).

The collected knowledge is combined to produce a classifier that helps estimate whether p is likely to get classified to the successful group or not. The resultant projection direction is derived as the weighted sum of all collected projection direction:

$$v_i = \sum_j^{n_i} w_j v_j \quad (\text{Eq. 4.18})$$

Similarly, the resultant centroid of successful group and unsuccessful group are calculated as:

$$c_{s,i} = \sum_j^{n_i} w_j c_{s,j} \quad (\text{Eq. 4.19})$$

$$c_{u,i} = \sum_j^{n_i} w_j c_{u,j} \quad (\text{Eq. 4.20})$$

The weight w_j for the j th knowledge provider is determined by its trust score:

$$w_j = \frac{t_j^i}{\sum_j^{n_i} t_j^i} \quad (\text{Eq. 4.21})$$

After combining other knowledge providers' projection direction (v_i), a_i transforms the combined centroid ($c_{s,i}, c_{u,i}$) using v_i . Similarly, the potential transaction p is also transformed and then classified using Euclidean distance of the transformed p to each transformed centroid. The distances are computed using Eq. 4.11 and Eq. 4.12.

4.5.2 LKSON Update

Since agents's behaviors may change overtime (e.g., honest knowledge provider may provide fake information or they may change their perspective on the system thus making the shared knowledge not useful to the requester any more, etc.), it is essential to update trust scores of the knowledge providers such that the inaccurate/useless knowledge has less impact on the final decision.

It is effective if we adapt trust scores of the knowledge providers accordingly every time a transaction is completed. However, such strategy is inefficient because it may incur unnecessary computation overheads. So we adapt a strategy by trading off computation overhead and performance, that is, after one transaction, if its outcome is unsuccessful, the requester a_i evaluates usefulness of local knowledge of each knowledge provider. For knowledge provider a_j^i , a_i simply uses v_j as the projection direction to calculate distances of transformed potential transaction from the transformed centroid (see Eq. 4.11 and Eq. 4.12). If the transaction is predicted as successful, it means the corresponding knowledge is incorrect or useless to a_i so trust score of the knowledge provider a_j^i must be lowered such that it has less impact on future decision. Otherwise, the trust score should be raised. The trust score update follows a Bayesian approach [19], which takes binary ratings as input and is based on computing trust scores by statistically updating beta probability density function (PDF). The posteriori trust score is computed by combining a priori trust score with new evidence. For knowledge provider a_j^i , we denote numbers of its successful and unsuccessful predictions by s_j^i and u_j^i respectively (i.e., the original trust score t_j^i is $\frac{s_j^i+1}{s_j^i+u_j^i+2}$). So a_j^i 's trust score is updated after one transaction using Eq. 4.22 (the new prediction is correct) and Eq. 4.23 (the new prediction is incorrect).

$$t_j^i = \frac{s_j^i + 2}{s_j^i + u_j^i + 3} \quad (\text{Eq. 4.22})$$

$$t_j^i = \frac{s_j^i + 1}{s_j^i + u_j^i + 3} \quad (\text{Eq. 4.23})$$

On the other hand, if the transaction is successful, the requester will evaluate usefulness of a small fraction (e.g., 20%) of the third party knowledge. This can help (1) avoid incurring too much unnecessary computation overheads because if the transaction is successful, it means combination of current third party knowledge is accurate enough to make a good prediction so no update is needed urgently and (2) proactively adjust weights of the third party knowledge thus making the combined knowledge more accurate (for the next prediction) while keeping the computation overheads low.

4.6 Conclusion

In order to improve StereoTrust by distinguishing useful stereotypes from the useless ones, we propose a machine learning based trust framework MetaTrust, which provide classification of a potential transaction by learning a set of a trustor’s past transactions (with other agents). In this work we instantiate the framework using two common machine learning algorithms, i.e., linear discriminant analysis and decision tree.

In case that a trustor does not have sufficient local knowledge, we propose to construct a local knowledge sharing overlay network (LKSON) to exchange agents’ local knowledge (i.e., intermediate results of classification algorithms) to predict trustworthiness of a potential transaction. Compared to traditional feedback aggregation based trust and reputation systems, such mechanism helps (1) reduce the possibility of sharing fake information; (2) avoid privacy leakage, (3) save a lot of computation and (4) avoid high communication overhead. This makes MetaTrust, compared to StereoTrust, applicable to a broader range of application scenarios.

MetaTrust is evaluated in the context of online auction (i.e., detecting fraud auctions by unknown sellers). On the other hand, StereoTrust is also modified and applied in the same context. But due to chronological order of our research done, StereoTrust has not been compared with MetaTrust so far. In order to conduct such a comparison, as well as to avoid repeated evaluation of MetaTrust in different chapters, we delay the evaluation part to Chapter 6 where we compare MetaTrust with StereoTrust as well as existing representative trust mechanisms in online auction.

MetaTrust assumes that the target agent’s behavior is static, which, however, does not always hold in reality. For instance, in multi-agent systems, a malicious agent may strategically change his behavior to maximize profits (and hence may harm interests of other agents). Such an issue will be partly addressed in Part IV of this thesis, but before that, we next present how to apply StereoTrust and MetaTrust to specific application scenarios to validate their practicability.

Publication(s)

- *MetaTrust: Discriminant Analysis of Local Information for Global Trust Assessment* (extended abstract),

Liu Xin, Gilles Tredan, Anwitaman Datta

In Proceedings of the 10th International Conference on Autonomous Agents & Multiagent Systems (AAMAS), Taiwan, 2011.

- *A generic trust framework for large-scale open systems using machine learning* (submitted to a journal, available on <http://arxiv.org/abs/1103.0086>). **Liu Xin**, Gilles Tredan, Anwitaman Datta

Part III

Applications of stereotyping based trust

In the previous part, we presented novel trust mechanisms that incorporate contextual features in trust assessment. In order to validate practicability of the proposed trust mechanisms and demonstrate their generic nature, we apply them to two distinct application scenarios. In Chapter 5, we first discuss how StereoTrust model can help address the data placement issue in the context of Peer-to-Peer (P2P) backup storage systems. Rather than addressing malicious peers, in this work, we focus on improving system reliability aided by contextual trust. Another application of the proposed trust models is to detect fraudulent behavior in online auction sites, which we present in Chapter 6. In such an application, our trust mechanisms are used to model behavior of the malicious sellers without their historical information. Note that due to chronological order of our research works, only StereoTrust is applied to P2P backup storage; both StereoTrust and MetaTrust are applied to online auction.

These two distinct applications demonstrate not only the fact that the proposed trust mechanisms are generic, but also the manner we apply StereoTrust for P2P backup storage (as an alternative to multi-objective optimization) furthermore demonstrates how trust based abstraction can be used to design robust distributed systems, where the trust is not being used to mitigate malicious behavior or intent, but as an abstraction to learn and adapt the system, without explicitly attempting to formulate or compute objective functions, which may be difficult in large-scale systems, either because of the absence of global knowledge, or any centralized coordinator.

Furthermore, the application of our trust mechanisms to the traditional realm - discerning malicious users in auction site, demonstrates our mechanisms' applicability as a potential alternative (or supplement) to traditional reputation based trust models.

Chapter 5

Enhancing Data Availability for Peer-to-Peer Backup

In Chapter 3, we proposed StereoTrust model, which is particularly designed to derive trust without the target agent’s historical information. We believe such a situation is very common in a large-scale, open distributed systems where an agent may frequently encounter strangers whose past behaviors are unknown to the assessor, even to the rest of the system. Although StereoTrust is already evaluated using Epinions dataset (see Chapter 3.3), we wish to further validate practicability of StereoTrust in a more complex and decentralized environment. To that end, we choose P2P backup storage system, which is large-scale, open and highly dynamic as the testbed.

P2P storage services are a cost-effective alternative for data backup. A basic question that arises in the design of such systems is: In which peers do we store redundant data? To address this issue, existing systems apply different techniques, including random selection, based on a distributed hash table (DHT) or based on the peers’ past availability pattern. In this Chapter, we propose as an alternative, a contextual trust based data placement scheme to select suitable data holders [92] (i.e., the ones provide high availability and fast transfer rate). It is originally designed for and applicable to scenarios where there is inadequate historical information about peers. Specifically, our scheme estimates trustworthiness of a peer based on stereotypes, formed by aggregating information of interactions with other (similar) peers.

Such an application demonstrates that besides modeling an agent’s trust relationship with other agents (for mitigating malicious behavior) just like traditional trust mechanisms, StereoTrust can also be applied to aid in robust distributed system designs. It can be treated as a complementary mechanism or an alternative to system’s multi-objective optimization, which may be nontrivial in large-scale systems.

The rest of the chapter is organized as follows. After the introduction in Section 5.1, we discuss background information in Section 5.2. We then elaborate our contextual trust based placement scheme in Section 5.3. Experiments are conducted to evaluate performance of our approach in Section 5.4 using real, as well as synthetic data-sets. Section 5.5 reviews related works, particularly with respect to existing data placement strategies for P2P storage systems, and Section 5.6 concludes this chapter.

5.1 Introduction

Traditional online backup solutions rely on dedicated servers for data storage. Given the multiple-fold increase in common users’ storage needs (personal photos, videos, etc.), commercial services are often prohibitive in terms of their monetary value. However, the growth of storage capacity, processing resources, and bandwidth available at the disposal of common users provide an alternative P2P paradigm [49, 50, 51, 53, 52, 100] for storing massive volumes of data. Such schemes amortize the participating users’ resources to provide a cost-effective alternative to commercial services.

While promising, designing a P2P backup system opens up various important issues including security, privacy and availability. The autonomy of the users and the scale of the system implies a high level of dynamics, and without global knowledge or coordination, achieving acceptable quality of service (QoS) is difficult. In the context of P2P backup, good QoS may refer to *ensuring high data availability (i.e., data is stored persistently in the network and can be accessed/updated at reasonable download/upload speeds) while minimizing resource consumption (e.g., bandwidth, storage)*. It is worth distinguishing between data persistence (the data is not irremediably lost) and data

availability (the data is accessible), which are two similar but distinct properties. Data persistence depends on long-term node behavior, whereas data availability is mainly determined by the current node behavior [61]. In this work, we focus on availability, rather than persistence. Availability is achieved using redundancy, e.g., replication or coding techniques [101, 102]. Data redundancy is lost over time due to system dynamics (e.g., correlated failures and permanent churn¹), thus compromising data availability. Various redundancy maintenance mechanisms [54, 103, 104, 105, 106] are proposed to generate new data redundancy to provide long-term data availability. A related (but complementary issue) is to determine the peers where such data should actually be placed. This chapter applies StereoTrust, utilizing easier to obtain meta-information, in contrast to existing approaches that use hard to obtain individual peers' historical information to guide the choice of data placement.

5.1.1 Motivation

Current data placement schemes for P2P storage can be broadly classified into two categories: (1) *local schemes*, for example, the ones relying on distributed hash table (DHT) [57, 58] to maintain data redundancy, or using only resources of social contacts [63, 107] and (2) *global schemes* that rely on random selection [56] or global knowledge like peers' historical availability [61, 62].

Local schemes such as the ones using DHTs, are very restrictive and do not leave much scope for system optimization at the storage layer. Global schemes are generally impractical when a peer is required to interact with unknown peers whose historical information is unavailable or unreliable. We propose and explore the efficacy of a new data placement scheme aided by contextual trust that is applicable in large-scale decentralized settings. Different from many existing trust models that rely on direct experience [31] and/or indirect experience [19, 26] to derive trust, our scheme is designed for scenarios where direct experience is unavailable and indirect experience may be unreliable.

---

¹Peers leave the system permanently.

Moreover, we note that even if a peer’s historical information is available through indirect sources, unlike traditional scenarios, “good” and “bad” may have somewhat different interpretations in a backup system application. Peers may inherently be bad - e.g., polluting content, free-riding, etc., and such behavior can be modeled using traditional reputation based trust models. However, peers may also be inherently well-behaved, but may be suitable for some peers and yet be unsuitable for some others. For example, even if a peer in New York is online for long hours, it may have negligible intersecting uptime with a peer from Hong Kong, while having a reasonably long uptime for peers from Europe. Thus, if its rating from these European peers is aggregated, or if a transitive trust model is used - assuming peers from Europe will be rated reasonably by peers in Hong Kong - the peer in New York will look trustworthy to the peer in Hong Kong, even though it is not a suitable candidate to store its data. This simplistic scenario provides motivations for using personalized, contextual meta-information based models to predict a peer’s suitability to carry out a specific task (and in this work that means, storing data).

5.1.2 The proposed scheme

We call a peer who has generated data redundancy and wishes to select *data holders* to store the redundancy as a *data owner*. A data owner estimates trust of an (unknown) potential data holder using stereotypes [44] that are learned from interactions with other peers. Thus, our work is based on the premise that a group’s trustworthiness (i.e., trustor’s stereotype of that group) can reflect its member’s trustworthiness.

To build stereotypes, a data owner groups other peers it has already interacted with according to some features taken from the meta-information of the peers, e.g., Internet Service Provider (ISP), time zone, connection type, etc. Note that these groupings are not exclusive of each other, and also are a personal perspective of each data owner. When facing a potential data holder, a data owner estimates its trust using the local stereotypes on the groups to which the potential data holder belongs. For instance, when a data

owner in Hong Kong wants to select data holders to store redundant data, by studying its past experience, it may find that peers in time zone GMT+8 (e.g., Singapore, Beijing, etc.) have a more common online pattern than peers in other time zones, say, GMT-6 (e.g., Chicago) and GMT+1 (e.g., Paris). Likewise, peers in Singapore and London may provide a fast connection speed while others do not. So when facing two unknown potential data holders who are in Singapore and New York respectively, the data owner would be able to make a first guess that the peer in Singapore is likely to be a better choice in terms of peer availability and connection speed.

5.1.3 Scope

Our scheme is designed as a practical light weight but sub-optimal alternative to existing data placement heuristics in decentralized settings. We note that a practical system may additionally use other supplementary mechanisms to further optimize the data placement decision. Such optimization, or hybrid models using both stereotypes and historical information, are beyond the scope of this work. Here we want to only investigate the efficacy of a contextual information based placement strategy.

One precondition of using our proposal is that the data owner must have some locally obtained knowledge. In practice, a completely new node will not have any local knowledge, and it may start with a randomized strategy. Alternative ways to bootstrap a node may include obtaining knowledge from other peers. In this work, we ignore such alternatives.

Furthermore, our scheme does not address malicious peers explicitly but focuses on a suitable data holder selection to improve QoS. In that sense, this work may be seen as a trust based alternative to system optimization techniques. Also, since our scheme is particularly developed for selecting suitable data holders from a pool of unknown peers, we do not discuss how to maintain trust relationships with known peers, which can be easily handled by a conventional trust mechanism. Some other issues in P2P backup such as search, node management, security concerns, etc. may influence data placement, but are also out of the scope of this work.

5.1.4 Summary

The novelty and practicality of our proposed redundant data placement scheme for P2P backup storage systems can be summarized as: (1) we adapt a contextual trust model [44] to design a new data placement scheme for P2P backup storage systems. Our scheme is particularly applicable to select suitable data holders when their global historical information is not available, a situation that is very common yet often ignored, and not comprehensively addressed in related literature. (2) Our scheme is generic and complementary to many existing approaches. It can be integrated into existing P2P backup systems that use global placement schemes. (3) We validate experimentally the effectiveness and efficiency of the proposed contextual trust based data placement scheme in comparison to random selection, as well as placement relying on the nodes' historical availability. By selecting suitable data holders considering time zone and ISP, our scheme provides higher file availability with judicious bandwidth usage, and the storage loads are reasonably balanced across all peers.

5.2 Background

Before describing the contextual trust based placement model for P2P backup storage systems, we briefly provide some background information on the system model for such storage systems, as well as trust modeling that we will use later to derive contextual trust for appropriate data holder selection.

5.2.1 System model

In P2P backup systems, a data owner (denoted by p_o) is a peer who owns or maintains data objects. It is necessary to store objects with some redundancy in order to achieve fault tolerance and provide availability even when some storage nodes are unavailable. Broadly speaking, two approaches can be used to realize such redundancy: replication and coding techniques, e.g., erasure coding [101, 102]. Erasure coding achieves high availability in a storage efficient manner, and is used in several P2P storage systems

[108, 54]. Throughout this chapter, we assume that M/N erasure coding is used such that N distinct encoded fragments for a data object O are stored and retrieving any M (distinct) fragments ensures reconstruction of the original data object. Then, a set of peers in the network are requested to store the N encoded fragments. We call these peers data holders (denoted by p_h). P2P backup system designs assume that any data holder stores (at most) one fragment of a data object. Otherwise, a single peer's failure will make multiple fragments unavailable, defeating the purpose of employing an erasure coding technique. To retrieve the data object at a later time, p_o contacts the data holders who are supposed to have stored the requested data fragments. When at least M distinct encoded fragments are received, the original data object can be reconstructed. We call this process of retrieving the data as a *transaction*. We denote that the set of transactions between p_o and p_h by Θ_{p_o, p_h} , and $\theta_{p_o, p_h} = |\Theta_{p_o, p_h}|$ denotes the number of such transactions. The notations are summarized in Table 8.1.

Failure of a transaction may occur as a result of various reasons. For instance, (i) the data holder p_h may be unavailable at the time of request, and thus does not respond; (ii) p_h is a free-rider and refuses to process the request; (iii) p_h deliberately sends counterfeit or corrupted content (pollution attack); (iv) the transfer rate is so slow that p_o gives up.

It is worth noting that in a more generic P2P storage system, a peer who requests a certain data object is referred to as a data requestor (denoted by p_r). A data requestor can be the data owner itself or other peers who have the right to access the data. But in the domain of P2P backup, only the data owner itself requests its own data (i.e., $p_r = p_o$).

5.2.2 Modeling trust

Modeling trust using beta distribution is a very popular approach [19, 31, 80]. Since outcome of a transaction is binary (successful or unsuccessful), we model the transactions between p_o and p_h as observations of independent Bernoulli trials. In each trial, the success probability, that is, the trust p is modeled by beta distribution. Please refer to

Table 5.1: Notations summary

Notations	Description
$\mathcal{P}$	Set of all peers in the system
$\mathcal{P}_x$	Set of peers that are known to peer p_x
p_o	Data owner who owns/maintains the original data object
p_h	Data holder who stores the fragments of data object of other peers to provide data availability
p_r	Data requester who requests a certain data object
O	A data object
N	The initial number of encoded fragments for each data object
M	The required number of fragments to reconstruct the original data object
Θ_{p_o, p_h}	Set of transactions between data owner p_o and data holder p_h
θ_{p_o, p_h}	Number of Θ_{p_o, p_h}

the original StereoTrust (see Chapter 3) for complete details on modeling trust using stereotypical groups.

We next investigate how contextual trust is calculated using our defined trust function (see Chapter 3.2.2) by considering contextual meta-information and how it can be used to choose suitable data holders in P2P backup.

5.3 Contextual trust based data placement scheme

5.3.1 Preliminaries

According to our contextual trust based data placement scheme (to be elaborated on later), data owner p_o selects suitable data holders based on peers' meta-information and p_o 's local knowledge. Normally, by carefully investigating characteristics of the potential data holder (e.g., resolving IP address), p_o is able to collect various types of contextual information such as the type of IP address, country/city/street, time zone, Internet service provider, operating system, etc. of the potential data holders. The approach of using contextual trust for redundancy placement is generic in that (m)any kinds of context may be used when determining the contextual trust, but some contexts may be more useful than others, while some others may not be useful at all. For example, peers' time-zones and Internet Service provider (ISP) may have a relevant impact on

the system's performance, while the operating system they run may not (or maybe it does, but is nevertheless not clear or observable).

For the validation of the efficacy of the approach, as well as for a more concrete elaboration of the ideas, we consider some specific kinds of contexts. More precisely, we consider peers' time zone as a context, since many measurement studies have demonstrated strong geographic correlation in diurnal patterns of peers' uptime [109, 110]. Data holders in the same time zone will demonstrate similar online patterns thus providing more effective availability for the data owner². Likewise, peers within the same ISP typically experience higher throughput at an individual level, and this also confines most of the network traffic within a single autonomous system, providing prudent usage of system wide network resources. Note that other contextual information such as the type of IP address (a node with public/static IP probably provides a more reliable connection than that with a private/dynamic IP), or a node's specific location (e.g., generally, nodes on a university campus or in work places are well connected to the Internet and stay online regularly and for extended periods as compared to home users) can also be used in our data placement scheme though such information have not been used in the reported experiments.

5.3.2 Local knowledge management

Our scheme estimates contextual trust of an unknown potential data holder using a data owner's local information such as its own past transactions and contextual meta-information of the peers that the data owner has already interacted with. In the system, each peer maintains a transaction database (TD) that records the experience of the data owner p_o with other peers. TD stores transaction information such as the identity of the transaction partner, the outcome of the transaction, etc. Table 5.2 gives an example of the structure of TD managed by data owner p_o .

---

²Note that for backup storage application, typically only the data owner needs access to the data. However, if multiple users are to access the data round the clock, storing at peers with different time-zones is desirable [111]. The proposed approach can be applied for such a scenario as well, but then the function needs to be changed.

Table 5.2: Structure of Transaction Database of Data Owner p_o

Tran. partner ID	Tran. ID	Tran. outcome
p_1	$T_{p_1}^1$	successful
p_1	$T_{p_1}^2$	unsuccessful
p_2	$T_{p_2}^1$	unsuccessful
p_2	$T_{p_2}^2$	successful
p_2	$T_{p_2}^3$	unsuccessful
.....		

Each peer also maintains another database: meta-information database (MD) that records the contextual meta-information of the data holders that this peer has interacted with, such as ISP, time zone, etc. Table 5.3 gives an example of the structure of a MD managed by data owner p_o .

Table 5.3: Structure of Meta-information Database of Data Owner p_o

Tran. partner ID	Location	Time Zone	ISP	...
p_1	Singapore	GMT+08:00	NTUNET	...
p_2	France	GMT+01:00	Dialer.net	...
p_3	Canada	GMT-05:00	BlastSurf.net	...
.....				

Using peers' contextual meta-information, the peers that data owner p_o has already interacted with (i.e., $\mathcal{P}_{p_o}$) are classified into different groups. We define a group as a set of peers that share similar properties or behave similarly. For instance, the peers who are in the same time zone can form a group because these peers may have the similar availability pattern; the peers who are administered by the same ISP can be grouped because these peers may share similar bandwidth. When facing an unknown potential data holder p_h , p_o first identifies its contextual meta-information to determine to which groups p_h belongs. Next we discuss how to obtain the potential data holder's contextual meta-information.

5.3.3 Collecting contextual meta-information about potential data holders

There are several ways to obtain an unknown data holder p_h 's contextual meta-information.

- Data owner p_o can learn p_h 's contextual meta-information by studying its IP address. This approach relies on certain geolocation databases to retrieve an IP's attributes such as country, state, city, ISP, latitude/longitude, telephone area code, etc. The caveat here is that geolocation technology is not 100% accurate in providing the location of an IP address. When the IP address is a proxy server address, it does not expose the user's IP address so it is virtually impossible to locate the user.
- When p_o cannot obtain p_h 's contextual meta-information by studying p_h 's IP address or the meta-information is not related to IP address, it may contact p_h directly. Here we assume that p_h has no motivation to send fake meta-information because it is unaware of p_o 's local transaction information. In other words, p_h is unable to mislead p_o by sending fake information, or the influence of fraud is quite limited and discontinuous. This is different from the traditional trust models where malicious nodes may intentionally send modified information to promote themselves.
- Another way to obtain p_h 's contextual meta-information is to contact other peers who may know p_h . We also assume that these requested peers have no motivation to send fake meta-information because they do not know p_o 's local knowledge thus cannot gain p_h 's relevance to p_o by sending fake data. Furthermore, in collaborative environments the trust mechanism is just trying to determine suitability of one node over the other to store data and there may be no/few malicious nodes intentionally subverting the system³. For the sake of data accuracy, p_o may contact multiple peers. If different sets of meta-information of p_h are returned, p_o just selects the set of meta-information that is provided by the most peers. This can help (1) reduce the probability of being cheated (in case that some peers send

---

³Dealing with and preventing such malicious behaviors will require other security mechanisms, possibly including conventional trust based approaches. That is an orthogonal issue, beyond the scope of the presented work, where we are using trust abstraction as an alternative to explicit multi-objective optimization of the system.

incorrect data); (2) estimate the most possible contextual meta-information of p_h (in case that p_h goes online from multiple locations. For instance, the user mainly goes online from a certain location (home/office), but sometimes she may also connect to the network from other locations (e.g., when traveling)).

5.3.4 Trust calculation

Once data holder p_o receives contextual meta-information of a potential data holder p_h , it can put p_h into the corresponding groups according to p_h 's meta-information (e.g., time zone, ISP, etc.). We denote these groups by $\mathcal{G}_{o,h} = \{G_{o,h}^1, G_{o,h}^2, \dots\}$ (for the sake of simplifying the notation, we will use G^i in place of $G_{o,h}^i$ when the context is clear).

Data owner p_o 's stereotype on group G^i is determined by p_o 's past experience with members of G^i (i.e., past transactions stored in p_o 's local transaction database (TD)). That is, p_o counts the total number of successful transactions S_{p_o, G^i} with G^i by summing up the successful transactions with G^i 's members: $S_{p_o, G^i} = \sum_{m \in G^i} S_{p_o, m}$. The total number of unsuccessful transactions U_{p_o, G^i} is computed similarly.

According to StereoTrust model, p_o can use SOF approach (Sum Of Functions) or SOP approach (Sum Of Parameters) to calculate and combine stereotypes (refer to Chapter 3.2.2 and Chapter 3.2.3 for trust function definition and trust calculation in StereoTrust):

$$TR_{o,h} = \sum_i W_{o,h}^i \cdot T_{p_o, G^i}(p | S_{p_o, G^i}, U_{p_o, G^i}), \quad (\text{Eq. 5.1})$$

$$TR_{o,h} = T_{o,h}(p | \sum_i W_{o,h}^i \cdot S_{p_o, G^i}, \sum_i W_{o,h}^i \cdot U_{p_o, G^i}), \quad (\text{Eq. 5.2})$$

where $W_{o,h}^i$ is the weight factor for group G^i (see Eq. 3.3 in Chapter 3.2.3 for weight calculation).

5.3.5 Algorithm outline: Selection of data holders

Consider the scenario where by applying any redundancy maintenance mechanism, data owner p_o generates R encoded fragments to increase availability (to a certain extent depending on which redundancy maintenance mechanism is used) of data object O . p_o

has to select R data holders from a set of unknown peers P_{un} to store the generated data. p_o first collects their meta-information about the peers in P_{un} as discussed in section 5.3.3 and calculates trust of these unknown potential data holders as discussed in section 5.3.4. Then based on trust ratings, p_o ranks these potential data holders and selects the R most trustworthy ones to store the fragments.

5.4 Evaluation

5.4.1 Simulation environment and methodology

In the experiments, we consider two kinds of contextual meta-information: (i) time zone, which influences peer availability pattern and (ii) location/ISP, which influences bandwidth (data transfer rate)⁴ among the peers. We assumed that data objects are stored as erasure encoded fragments, such that originally N distinct fragments are stored in the network and any M distinct fragments ensure file reconstruction. The ratio of M and N is the storage efficiency δ (M/N). Obviously, the higher the storage efficiency, the weaker the resilience against node failures. We will study the influence of δ in the experiments. Two kinds of storage efficiency are used: 8 out of 32 ($\delta = 0.25$) and 16 out of 32 ($\delta = 0.5$).

In practice, some peers in the network may leave permanently, which causes permanent loss of data stored on them. *Permanent churn* thus makes the system more vulnerable, and unless mitigated, may not only disrupt availability but also lead to permanent loss of stored objects. Various redundancy maintenance strategies are proposed to address permanent churn [54, 103, 104, 105]. We adapt *deterministic lazy repair strategy* [54] in the simulation. That is, when the number of encoded fragments for each file falls below a threshold T_{dl} , repair operations are triggered and new fragments are generated to make the number of encoded fragments reach back to N .

Two simulation environments are used: Environment A (using PlanetLab trace [112]) and Environment B (using synthetic trace).

---

⁴The data transfer rate is influenced by many factors such as bandwidth, connection type, ISP latency, etc. However, for simplicity, we assume that the data transfer rate is only determined by bandwidth in the simulation.

5.4.1.1 Environment A

PlanetLab trace records the availability pattern of Planetlab nodes from January, 2004 to June, 2005. The reasons that we use PlanetLab trace are: (1) it is a well-known trace that is commonly used to evaluate distributed systems [113, 114], (2) it provides IP addresses of the nodes that can be used to retrieve contextual meta-information of these nodes. Fig. 5.1.a shows the node availability (as well as permanent departures).

We use an online IP lookup service [115] to retrieve meta-information of these nodes. There are 669 nodes in the trace and only two of them cannot be looked up (we omit these two nodes in the simulation). Using the retrieved contextual meta-information, we can construct groups/stereotypes for our approach.

Ideally, ISP determines node bandwidth. However, in the trace, each ISP only administrates a very small number of nodes. To handle this, we use the node’s location (country level) instead of ISP to determine data transfer rates among nodes. To make the simulation environment more realistic, we further configure that the nodes in the same country are assigned bandwidths following normal distribution $[\mu, \sigma]$, where μ is set according to the country (e.g., in country a the μ is set as 500Kbps and in country b the μ is set as 100Kbps) and σ is set as $\mu/5$.

We configure that each node has three files and hence, $667 \times 3 \times 32$ data fragments as being the total load “ideally” if all fragments of all objects are online (and no duplication of encoded fragments is present).

5.1.a: PlanetLab trace

5.1.b: Synthetic trace

Figure 5.1: Node availability

5.4.1.2 Environment B

PlanetLab trace consists of only 669 nodes. Moreover, the nodes are dedicated computers supposed to be run 24/7, and do not reflect human temporal behavior. To overcome these shortcomings, we generate synthetic trace which simulates a large scale heterogeneous P2P network for a more comprehensive performance evaluation.

The P2P network topology is generated using BRITE [116] that has recently emerged as one of the most promising universal topology generators. We use the *2 Level: TOP-DOWN* type to generate a topology composed of 100 autonomous system (AS) and 10000 nodes. The AS level is firstly constructed according to one of the available flat AS-level models (e.g., Waxman [117], Imported File, etc.). Then for each node in the AS-level topology, BRITE generates a router-level topology. The final topology is generated by flattening the hierarchical topology into a router-level one composed of the individual topologies associated with each node at the AS-level. The size of the generated topology (plane) is 1000×1000 and each node is a point (represented by a 2D XY coordinate) on the plane. The nodes are interconnected using *Waxman* model [117] and the average number of links for each node is 3.

According to our generated network topology, each peer belongs to a certain AS. Normally, each AS is administered by an ISP that determines the bandwidth of peers in this AS. So the data transfer rate between a pair of peers depends on which AS they belong to. Note that to make the simulation environment more realistic, the nodes under the same ISP are assigned bandwidths following normal distribution just like that in simulation environment *A* (see Section 5.4.1.1).

Several works [109, 110] have proven the diurnal pattern of peers' availability. We borrow the definition of a time dependent availability function from [110]:

$$f(t) = \begin{cases} (6 - t)/10, & \text{if } t < 6, \\ (t - 6)/14, & \text{if } 6 \leq t \leq 20, \\ (30 - t)/10, & \text{if } t > 20 \end{cases}$$

Where t is a variable representing the hours in a day.

In the network topology, each peer has a position (x, y) . We set the time zone of each peer according to its position (actually x-coordinate). That is, we treat the plane of the topology as a global map and divide it into 24 time zones equally. A peer is in a certain time zone if its position (x, y) falls into the area of this time zone. So the probability that a peer is available depends on its time zone, current time point, as well as its individual failure model. For instance, from simulation time 0 to 59, we set that local time of the peers in time zone 0 as 0am (each time unit stands for one minute), local time of the peers in time zone 1 is 1am, and so on; from simulation time 1560 to 1619, local time of the peers in time zone 0 is 2 a.m., local time of the peers in time zone 1 is 3 a.m. and so on.

Besides time of day effects, each peer has a unique failure behavior which depends on various factors such as aging of the software/hardware, available resources, etc. Stutzbach et al. [118] studied two fundamental properties of the churn: (i) inter-arrival time, which is the time that passes from the start of one session⁵ to the start of the next session and (ii) session length, which is the time that passes from the start of a session until the end of that session. According to the conclusions of [118], we model inter-arrival time and session lengths of peers using Weibull distribution. We adjust shape and scale parameters to assign session lengths and inter-arrival time to individual nodes such that node availability within the same time zone is similar to that defined by time dependent availability function. Fig. 5.1.b shows the node availability in the synthetic trace and default parameter values⁶ are summarized in Table 5.4.

5.4.1.3 Performance metrics

We compare performance of different placement schemes using the following metrics:

(1) *File availability.* In P2P backup systems, data availability is influenced by many

---

⁵We define a session as a process that a peer joins the system, contributes/provides resources and leaves the system.

⁶We also tried other values for e.g., T_{dl} , N_l , $\mathcal{F}$, $\mathcal{T}$, etc. and observed the similar qualitative results.

Table 5.4: Simulation Parameters

Parameters	Description	Default Value
N	The initial number of encoded fragments for each object.	32
M	The required number of fragments to reconstruct the original object.	8 or 16
T_{dl}	The threshold number of fragments for deterministic lazy repair to trigger the repair operation.	12 or 24
N_{tz}	The number of time zones used in the simulation	24
N_l	The number of ISP used in the simulation	100
$\mathcal{N}$	The number of peers in the simulation.	10000
$\mathcal{F}$	The number of files owned by each peer	3
$\mathcal{T}$	The number of time units for each simulation running	10080

factors such as the redundancy strategy, redundancy maintenance mechanisms, as well as data placement schemes. In our experiments, we use the same redundancy strategy and redundancy maintenance mechanism to test the influence of various data placement schemes. (2) *Bandwidth consumption*. Generally, bandwidth consumption in a P2P backup system mainly depends on which redundancy maintenance strategy is adapted [119]. However, a good data placement scheme indeed helps reduce the probability of triggering a repair operation, thus reducing bandwidth consumption. (3) *Storage load*. Although users' storage capacity is becoming higher, it is unreasonable to design a large-scale distributed storage system without considering the balance of the storage load. Moreover, more data stored means potentially more data requests, so imbalance of the access load may be caused by imbalance of the storage load. Therefore, we also characterize the resulting storage load distribution for different placement strategies. (4) *Overhead for data access*. Finally, we also look at the overheads incurred for data access (i.e., average messages sent for retrieving file objects per data access), when applying different placement schemes. Such a metric helps us understand network complexity in terms of sent messages, i.e., efficiency of reconstructing the original data objects under different data placement schemes. Note that the bandwidth and storage are measured in terms of number of encoded fragments.

5.4.1.4 Alternative placement strategies

Our placement strategy is compared with respect to two other existing global placement strategies:

- *Random selection.* Such a scheme places encoded fragments on some randomly selected peers. Due to its simplicity, random selection has been widely used in many storage systems [54, 56].
- *Peer availability history based data placement scheme.* Such a scheme places encoded fragments on the peers according to peers' availability [61]. Specifically, at each time unit, we collect availability information of each peer, which is represented by an up-time score. If a peer is online, its up-time score is increased by one. The redundant data is placed on the peers with the high(est) up-time scores.

Note that peer availability based data placement schemes assume (accurate and complete) global information to choose data holders, which is unrealistic in large-scale distributed settings. However, it can be used as a reference, demonstrating how data availability will be if only the peer's historical availability is used.

Besides the placement strategies mentioned above, we also take into account what can be best achieved in a hypothetical setting, even the future behavior of nodes is known accurately, which we call the *optimal scheme*⁷. That is, we assume each data owner knows the accurate future online behavior of other peers as well as their data transfer rates so that it can make the optimal selection of data holders who demonstrate relatively high node availability in the future and provide satisfactory data transfer rates. Such an optimal scheme determines what is the best that can be done. Since it needs the knowledge of the future, it is not feasible in practice, but provides us another baseline to compare our strategy. Note that no other strategy, even by chance, can excel this hypothetical optimal scheme in terms of data availability.

---

⁷Note that in a real PlanetLab trace, node's online pattern is already known, while in synthetic trace, node's availability is generated artificially according to its time zones, time of day effects, and unique failure model characterized by the inter-arrival time and session length (see Section 5.4.1.2) and is hence, again known a priori for the experiments.

5.4.2 Results

In our previous work [44], we observed that when calculating trust, SOF approach is slightly more accurate than SOP approach (see section 5.3.4). In this work, we also did experiments to compare these two approaches and found that SOF indeed performs better even in this specific application scenario. So in the rest of this chapter, we report results where the data placement scheme uses SOF approach by default unless otherwise specified. Each experiment is repeated ten times to ascertain that the observed results are consistent.

5.4.2.1 Environment A (PlanetLab trace)

We first compare the placement schemes with an 8 out of 32 ($\delta = 0.25$) erasure code scheme being used. We begin with a comparison of all placement schemes in terms of file availability. We only show results once the system has moved into a steady state. From Fig. 5.2.a, we can see that our scheme outperforms both random selection and the peer availability based scheme. Particularly when correlated failures occur (at time 7700 a 72.5% of all nodes go offline simultaneously), our scheme still maintains over 75% file availability while file availability for the peer availability based scheme falls below 60%. This is because our scheme considers not only peer availability, but also other aspects like download speed (if peers cannot tolerate the low download speed, it gives up and considers the data is unavailable) to improve user experience of accessing stored data. Additionally, both random placement and historical uptime based schemes naturally favor some specific nodes (with high availability) where many data items are placed, and hence, lost together once a correlated fault occurs. This explanation is corroborated from the storage load distribution results in Fig. 5.2.d. The optimal scheme is clearly the best placement scheme that maintains higher file availability throughout the whole simulation (over 92% file availability is achieved even when correlated failures occur at time 7700). Fig. 5.2.b measures the file availability in another manner, that is CDF of percentage of online time (the ratio of time points that a file is available divided by all

time points) per file. We observe that our scheme leads to higher percentage of online time of stored objects in comparison to the random selection and peer availability based schemes.

5.2.a: File availability

5.2.b: File availability (CDF)

5.2.c: Bandwidth consumption

5.2.d: Storage load

 Figure 5.2: Real PlanetLab trace ($M = 8$, $N = 32$)

Now we measure bandwidth consumption of the placement schemes. We observe from Fig 5.2.c that the optimal scheme consumes the least bandwidth due to its accurate selection of data holders. Our scheme consumes less bandwidth than random selection and peer availability based schemes do. This is because our scheme helps choose the suitable data holders that make the data more available and thus, less fragments are generated for replacement and less bandwidth is consumed. We also observe that in some periods (when correlated failures occur), the bandwidth usage for the peer availability based scheme is quite spiky. This is because this scheme is more likely to place fragments on highly available peers. However, once these peers go offline, large amount of fragments are lost, which causes huge and spiky bandwidth usage during repairs.

We measure the storage load of these placement schemes. From Fig 5.2.d, we can see that random selection balances the storage load well because it places the fragments on randomly selected peers from a pool of candidates. Please note that in the long run, random selection also favors the nodes that are more likely to go online [120]. However, when the highly available nodes go offline or behave abnormally, new candidates are found and the data redundancy will be placed on these (online) candidates with the same probability. So overall, random selection scheme balances storage loads better than other schemes do. Our scheme performs similarly to random selection, thus also balancing the storage loads properly (storage loads of over 80% peers range from 120 to 200 and the highest storage load is 262). On the other hand, the peer availability based scheme suffers from a serious storage load imbalance: about 60 percent of peers have a storage load of less than 100; about 30 percent of peers have a storage load from 100 to 150 and about 10 percent of peers have a storage load larger than 150. Moreover, about 6% of peers's storage loads are over 1000.

Similar to a peer availability based scheme, the optimal scheme also suffers storage load imbalance issue because it prefers to store data redundancy to the peers with high availability. However, since the overall redundancy generated by optimal scheme is quite limited (i.e., optimal scheme chooses the most trustworthy data holders thus incurring less repair operations, and consequently the generated redundant data is less than that incurred by other schemes), the storage load imbalance issue is not as pronounced as with the peer availability based scheme.

Finally, in Fig. 5.3 we study data access overhead in terms of average messages to probe peers in order to retrieve an object, i.e., find M live nodes storing distinct encoded fragments of the object. The average number of messages over all successful object accesses is plotted and the error bar represents a 95% percentile. The results are qualitatively analogous to that of file availability and bandwidth consumption. Except for the optimal scheme, which obviously requires the least messages, our scheme uses less messages than the random selection and peer availability based schemes. This is

because, compared to the other two data placement schemes, our scheme maintains a higher file availability, the probability that a message is sent to a peer that can return the requested data is higher, i.e., less messages are needed to retrieve a certain file object. We also plot a horizontal line indicating the overhead that would have been incurred if there were no failures. This overhead is in fact M (8 in this scenario) in M/N erasure coding, since no additional probing is needed if all nodes are available.

Figure 5.3: Messages sent for retrieving required files (real PlanetLab trace: $M = 8$, $N = 32$).

We note that besides the overheads during data access to determine enough available peers/fragments (as discussed above), there are two other kinds of protocol overheads in the system: (i) The overheads in carrying out the data placement itself, i.e., how to determine the specific set of data holders. For our scheme, the data owner may resort to a centralized repository (e.g., geolocation databases) to resolve IP address to obtain meta-information (size of such a repository is independent of the network population, but dependent on the different kinds of meta-information being used). For an availability based scheme, the data owner also resorts to a centralized repository to collect potential data holders' past availability information (size of such a repository, however, depends on the size of the peer population). Nevertheless, for every data placement scheme, such overheads are marginal/negligible and thus are not modeled or evaluated in the simulation. (ii) The actual overheads of data transfer during data placement or access that are the same for all the schemes. Such overheads are in fact the dominant cost

since data objects are typically large files, thus making the other two kinds of overheads insignificant in comparison.

We continue to study and compare the performance of various data placement schemes when a different erasure code scheme ($M = 16$, $N = 32$) is used. From Fig 5.4.a and 5.4.b we observe that when comparing 8 out of 32 erasure code schemes, in this scenario, the file availability is relatively lower (even for the optimal scheme). This is because when M is bigger, more encoded fragments are needed to reconstruct a file object, so the probability that a file can be retrieved is lower. Also, due to the bigger value of T_{dl} , more repairs are triggered to replenish the lost fragments that consequently consume more bandwidth (Fig. 5.4.c). In this scenario, except for the optimal scheme, our scheme still provides higher file availability and consumes less bandwidth in a smooth manner (i.e., there is no sudden big increase of bandwidth usage that may overwhelm the network traffic). Moreover, the storage loads are distributed across all peers fairly (Fig. 5.4.d: storage loads of over 80% peers range from 120 to 300 and highest load is only 315). The peer availability based scheme provides higher file availability than random selection does but consumes more bandwidth in a spiky manner and its storage load imbalance issue is particularly serious (the highest storage load is as high as 4096).

5.4.2.2 Environment B (Synthetic trace)

Fig 5.5.a shows CDF of percentage of online time per file when 8 out of 32 erasure code schemes and synthetic data are used. As expected, the optimal scheme provides the highest file availability. Both the peer availability based scheme and the random selection perform quite similarly while our scheme outperforms both of them. We observe that for our scheme, around 73% of the files have more than 90% online times and only around 2.5% of the files have less than 50% online times. On the other hand, for the other two schemes, around 10% of the files have less than 50% online times, which means most of the time, these files are unavailable. Only about 60% of the files have over 90% online times.

5.4.a: File availability

5.4.b: File availability CDF

5.4.c: Bandwidth consumption

5.4.d: Storage load

 Figure 5.4: Real PlanetLab trace ($M = 16$, $N = 32$)

Since our scheme provides higher file availability, when data redundancy repair operations are triggered (if necessary), less fragments need to be generated compared to random selection and peer availability based schemes (Fig 5.5.b). So except for the optimal scheme, our scheme consumes the least bandwidth. On the other hand, the peer availability based scheme consumes the most bandwidth and the usage is quite spiky. Comparable to our scheme, random selection smoothly consumes the bandwidth, but the overall bandwidth usage is higher.

We also measure the inter-ISP and intra-ISP network traffic to demonstrate how network resources are used by each placement scheme (the traffic is measured in terms of the number of encoded fragments transferred). Peers within the same ISP typically experience higher throughput at individual levels thus offering improved user experience. This also confines most of the network traffic within a single autonomous system, providing prudent usage of system wide network resources. From Fig. 5.5.c we observe

5.5.a: File availability CDF

5.5.b: Bandwidth consumption

5.5.c: Inter/Intra-ISP network traffic

5.5.d: Storage load

 Figure 5.5: Synthetic trace ($M = 8$, $N = 32$)

that for our scheme, intra-ISP traffic dominates the total network traffic while for the other schemes, the difference between intra-ISP traffic and inter-ISP traffic is not very obvious because they do not take into account the peer's ISP when selecting data holder.

Finally, we compare storage loads caused by the placement schemes. As mentioned earlier, random selection balances storage loads well compared to other schemes and thus, is treated as the baseline. From Fig 5.5.d we observe that the curve of our scheme correlates well with that of random selection, demonstrating that the storage load is well balanced while using our scheme (most of the peers have storage loads ranging between 50 and 300). In contrast, the peer availability based scheme suffers serious storage load imbalance issues. A small fraction of peers have a storage load of over 300 and the maximum storage load is 29299 (not shown on the plot). The optimal scheme, on the other hand, also suffers storage load imbalance issues but not as seriously as the peer availability based scheme does.

 Figure 5.6: Synthetic trace ($M = 16$, $N = 32$)

When 16 out of 32 erasure code scheme is used, similar to environment A, higher storage efficiency δ makes file availability lower (Fig. 5.6.a). Moreover, more bandwidth is consumed (Fig. 5.6.b) due to the higher value of T_{dl} . In this scenario, except for the optimal scheme, our scheme still outperforms the other two schemes in terms of file availability and bandwidth usage. Similar to the previous scenario where 8 out of 32 erasure code scheme is used, we notice from Fig. 5.6.c that the intra-ISP traffic dominates the total network traffic generated by our scheme while the other three schemes do not (intra-ISP traffic is just slightly higher than inter-ISP traffic). For storage load distribution (Fig. 5.6.d), our scheme also performs quite similarly to random selection that balances the storage loads well across the peers.

So to sum up, our contextual trust based data placement scheme performs effectively and efficiently in different environments due to its ability of identifying and personalizing the most suitable data holders according to their individual characteristics

(meta-information) based on the data owners' local knowledge. It provides a practical alternative to the existing data placement strategies in P2P backup systems.

5.5 Related work

Storing massive data in a Peer-to-Peer manner is interesting academically as well as in practice. There are many aspects that need to be considered when designing such a system, and the crucial ones are summarized in Fig. 5.7. This section only reviews related works regarding one dimension, i.e., data placement issue.

Figure 5.7: Important design decisions that need to be considered when designing and deploying a P2P storage system

An example commercial P2P (hybrid) storage system is Wuala [121], which uses a mix of dedicated infrastructure as well as resources from participating peers. Fig. 5.8 shows the architecture of Wuala. The dedicated infrastructure is used for bootstrapping and as a fall-back option. There are two types of peers in the network: (1) super peers that participate in a DHT for indexing fragment location and (2) storage peers

that provide storage resource and link to the super peers. Wuala relies on symmetric exchanges of data between users. Files are split up into (encrypted) fragments and spread in the network. Super peers are responsible for distributing the data on the storage peers uniformly. When a user requests a certain file, Wuala first locates the fragments using a distributed hash table (DHT). The query is routed through super peers. Then the fragments are downloaded from multiple storage peers simultaneously to construct the requested file. Wuala offers an easy way to store data in a P2P manner. Users' data is processed and distributed in the network transparently to the users. Unfortunately, there does not exist much information about the technical aspects of the Wuala network like probability of data loss, user population, data location, etc.

Figure 5.8: Wuala architecture

In academia, one of the most widely used approach is to determine the placement of objects based on keys. In this scheme, all the peers as well as data objects are assigned unique keys (e.g., by hashing). Then the data objects are stored at the peers with closest or similar keys. The peers are communicated among each other by forming a structured overlay network, and the messages are routed based on key similarity. Such an object placement scheme can be seen in [122, 108, 104, 123] to name a few. This approach essentially combines the storage of the data with the search mechanisms. The shortcomings of this scheme are: (1) drawbacks of DHT itself [75] (e.g., it does not

work well in a highly dynamic network), (2) users have no control over the choice of placement (e.g., unsuitable peers may be selected thus causing low data availability and high overheads) (3) it cannot be used by other unstructured P2P networks.

Alternatively, data placement may be decoupled from the search mechanisms, thus giving more flexibility in choosing the storage locations. This choice may be random [54, 55, 56], or based on peers' historical information [61, 62, 124], or based on proximity (locality) from the end users accessing the data [125], or based on load at the peers, or other considerations like storing the data within a specific domain.

Giroire et al. [126] studied the impact of different data placement schemes on the system performance when using erasure codes. Three placement policies are compared: (1) Chain policy, which uses DHT to store fragments on logically closest neighbors. (2) Buddy policy, which stores fragments within a buddy group and (3) Global policy, which selects the data holders randomly. Simulation results show that when there are no constraints on bandwidth, the bandwidth consumption among the peers are distributed much more smoothly for global policy but the average bandwidth usage is the same for all three policies. However, chain policy and buddy policy have much longer mean time between data loss events. When the maximum available bandwidth is limited, the global policy performs better in terms of probability to lose data. Additionally, chain policy and buddy policy incur much more frequent data loss events compared to the scenario where bandwidth usage is limited.

Lluis et al. [100] proposed a distributed backup system that aims to improve data availability and network fairness by considering peer stability. Based on local knowledge (e.g., other peers' availability patterns), peers with the similar stabilities are automatically clustered where they choose partners to exchange data symmetrically for data backup. Such a strategy rewards stable peers with more availability and prevents unstable ones from gaining sufficient backup space. This also stimulates unstable peers to improve their stabilities and hence improves the overall availability/stability of the system.

The work [62] focused on two formal problems of availability matching in P2P networks: disconnection matching (finding peers expected to disconnect at the same time) and presence matching (finding peers expected to be online simultaneously in the future). Epidemic protocols [127] are used to solve such matching problems. The presence matching can be applied to find suitable data holders in P2P backup systems (i.e., the selected data holders are online when data owner itself is online). This is similar to the goal of our work. However, it requires rich historical availability information, which is not always available.

Bernard et al. [124] tried to optimize P2P backup by selecting suitable data holders using lifetime estimation. The authors believe that lifetime of peers in the system is highly correlated to the length of their presence in the systems. So when a data owner wants to store data on the network, it selects data holders based on their stability, which is measured by their ages: the longer a peer has been in the system, the more stable it will be considered. In this work, a secure monitor protocol for peer availability is assumed. However, it also requires rich historical availability information.

Li et al. [128] proposed to build a reliable P2P backup system by taking into account social trust (F2F). The basic assumption of this systems is that just as people interacting in the real world are much less likely to cheat friends than strangers, peers in F2F system also act cooperatively towards other peers if this user knows that these peers are controlled by his real world friends. Hence, there is no need to design any mechanisms (e.g., historical behavior based) to measure reliability of other peers. This assumption significantly simplifies the system design. Following F2F, Dinh et al. proposed Friend-store [63] to address challenges that are ignored by F2F: (1) ensuring that a friend peer will not store more data than it can maintain reliably and (2) trading off bandwidth for storage when disk space is a more limiting resource. To address the first challenge, the authors proposed to let each peer in the system calculate its data maintenance capacity based on its upload bandwidth. Then the amount of data stored by each peer is limited accordingly. To address the second challenge, the authors introduced a coding scheme

XOR(1,2) to make a data holder stores coded replicas for multiple users simultaneously. That is, more bandwidth will be consumed during the restore in order to reconstruct the original data.

5.6 Conclusion

In this chapter, we validate practicality of StereoTrust by proposing a stereotyping based data placement strategy for P2P storage systems. Different from many existing data placement schemes that rely on peers' historical information, our scheme is particularly designed for the scenario where a specific peer's online pattern is not available by using different kinds of information (based on stereotypes) to estimate contextual trust (i.e., suitability) of unknown potential data holders. To build stereotypes, data owner groups other peers it has already interacted with according to some features taken from contextual meta-information of the peers (like ISP, time zone, connection type, etc.). Then, when facing an unknown peer, the data owner estimates trust according to its stereotypes on the groups to which the unknown peer belongs.

We evaluate performance of the proposed placement scheme using PlanetLab trace as well as synthetic trace. Results show that under different environments, our scheme provides higher file availability than schemes based on the peer availability and random selection, while also consuming bandwidth judiciously (the bandwidth usage is the least and smoothest). Our scheme balances the storage load as good as random selection while the peer availability based scheme suffers serious storage load imbalance issues. Moreover, our scheme confines most of the network traffic within autonomous systems, providing prudent usage of system wide network resources.

The application of StereoTrust for data placement in P2P storage systems demonstrates that StereoTrust can be used not only for security (e.g., mitigating intentional/malicious behaviors) like traditional trust management mechanisms, but also for improving systems' reliability and performance where StereoTrust can be viewed as an abstraction to learn and adapt the system without explicitly formulating or computing objective functions, which may be difficult in large-scale decentralized systems.

Currently, this work only focuses on estimating trust of unknown peers without the historical information of the specific peer in question. In the future, we expect to extend current approach to improve prediction accuracy by taking into account some global information.

Publication(s)

- *Contextual trust aided enhancement of data availability in peer-to-peer backup storage system,*

Liu Xin, Anwitaman Datta

Journal of Network and Systems Management [Springer], 2011 (published online).

Chapter 6

Identifying Risky Transactions in Internet Auctions

In this chapter, we continue validating practicality of our trust mechanisms in another application scenario, namely, online auction sites. The goal of this work is to detect the potential auction frauds caused by malicious sellers. Specifically, we focus on the scenario where a potential seller is unknown to the buyer, and even more restrictively, unknown to the rest of the system (i.e., a buyer cannot estimate trustworthiness of the unknown seller based on the seller's past behavior). Our approach first identifies discriminating attributes which are capable of distinguishing successful transactions from unsuccessful ones. A buyer can then use its own past transactions (with other sellers) to form stereotypes according to the identified attributes. Both StereoTrust and MetaTrust are used to derive stereotypes based trust [129].

This chapter is organized as: after introduction in Section 6.1, we discuss our approach for detecting fraud auctions using MetaTrust in Section 6.2 and StereoTrust in Section 6.3 respectively. Note that in order to adapt to online auction environment, we used a modified version of StereoTrust. In Section 6.4, we conduct experiments using real auction dataset from Allegro as well as synthetic datasets to evaluate performance of the proposed approaches.

6.1 Introduction

Online auction (e.g., eBay [2], Taobao [130], Allegro [64], etc.) constitutes one of the most successful Internet business models. Users from all over the world trade goods worth millions of dollars every day using these virtual marketplaces. For an online auction site, one of the most important issues is to ensure high quality of service for the buyers, that is, to support them by selecting trusted sellers. Unfortunately, rapid commercial success has made auction sites a lucrative medium for committing fraud. The Internet Crime Complaint Center [131] reported that Internet Auction Fraud was by far the most reported offense in 2007, comprising of 35.7% of all complaints. In addition, during the same period, the non-delivery of merchandise and/or payment from non-auction Internet transactions represented another 24.9%, pegging the combined total of online sales related fraud complaints at 60.6% of all Internet fraud. So security mechanisms such as trust and reputation management are needed to determine risks by predicting potential malbehavior, and provide decision support to end-users.

However, despite frequent criticisms, only the simplest reputation systems (i.e., aggregated ratings and comments) are used by the most popular Internet auctions today because Internet auction providers are generally not willing to modify current reputation system infrastructure. As a consequence, an experienced auction buyer is forced to undergo the menial task of reading and judging comments about his potential transaction partners [132]. While human judgement is the best possible way of evaluating this information, the task is time-consuming and error-prone: the sheer number of comments is sometimes an obstacle to making a good decision under uncertainty. On the other hand, the relatively inexperienced buyer becomes an easy target of frauds by being unable to deal with the scattered information.

Many previous solutions [133, 19, 134, 80, 135, 136, 81] endeavor to help inexperienced buyers detect auction frauds by relying on specific sellers' past behavior. However, such information is not always available or may be insufficient. For instance, the seller newly joins the system or sells items infrequently. However, often the kind of fraudulent

behaviors and scams are repeated - by the same sellers, as well as across different sellers [137].

We propose a stereotyping based approach to estimate trustworthiness of a potential transaction where the seller is unknown to the buyer, i.e., the buyer has no or little historical information of this seller to study. Our approach builds upon some recent advances: (1) previously proposed StereoTrust and MetaTrust, which utilizes meta-information based stereotypes to estimate trust of a seller whose past behavior is not available, and (2) a web browser plug-in called ProtoTrust [138] to crawl auction sites and use a library of new and existing algorithms and decision making rules to detect frauds. Note that ProtoTrust was developed separately by Kaszuba et al. and is in itself not a contribution of this work.

The novelty of this work is the use of stereotypes derived from meta-information, instead of specific sellers' historic information (alternatively, complementing it) leveraging on the fact that auction frauds often have some common attributes and thus can be stereotyped. At a high level, the approach is analogous to spam email detection mechanisms. Also, it bears resemblance with collaborative filtering techniques. The primary difference is that our approach uses only local information or other information that is publicly accessible in Internet auction sites.

In our approach, a buyer estimates trust of the potential transaction, which is conducted by an unknown seller by combining corresponding stereotypes. A stereotype is determined by attributes, which can be taken from the profiles of the sellers. To build stereotypes, the past transactions that a buyer knows are classified into different groups according to the attributes. For instance, the transactions which sell items in the category *Mobile Phone* are grouped. Two types of past transactions are used: the buyer's individual local knowledge (i.e., this buyer's past transactions with the sellers) and community knowledge (i.e., collection of other buyers' past transactions with the sellers). Then when facing a potential transaction, the buyer derives trust of this transaction using stereotypes on multiple such groups to which this transaction belongs. So,

to detect auction frauds, it is crucial to identify attributes whose values are capable of distinguishing successful transactions from unsuccessful transactions. By studying Allegro dataset, we identify three useful classes of attributes: (1) category of the item sold in the transaction, (2) price of the item and (3) number of items already sold by the seller when the transaction is conducted. We combine these three attributes to form the stereotypes.

The contribution of this chapter is threefold. First, we propose a stereotyping (StereoTrust/ MetaTrust) based approach for detecting auction frauds. This approach is particularly designed for the scenario that the seller of the potential transaction is unknown to the buyer. Please note that the goal of our approach is to identify whether the encountered transaction is good or not, so we do not discuss how to maintain trusts of the known sellers. Second, our approach is generic and complementary. It does not require any modifications of current online auction sites. It is being integrated into a library of trust management tools developed in the Universal Trust project (uTrust) [139] whose aim is to create standard trust management services for distributed, open systems. Third, experiments conducted over a large-scale, real dataset, as well as synthetic datasets shows that our approaches are effective to detect auction frauds while avoiding false negatives.

6.2 MetaTrust based approach

Recall that MetaTrust is a machine learning framework for trust assessment. It takes as input a set of attributes and the buyer's local knowledge or knowledge collected from other buyers. It then applies suitable machine learning algorithms (linear discriminant analysis and decision tree in this case) to predict whether the potential transaction is risky or not. The whole process is straightforward and automatic so we do not repeat it here. Please refer to Chapter 4 for more details. We next describe an alternative approach, which is based on modified StereoTrust.

6.3 StereoTrust based approach

In this section, we present our approach for detecting auction frauds based on modified StereoTrust. Specifically, for a particular group of past transactions, we divide them into two disjoint sub groups, each of which contains either successful transactions or unsuccessful ones. We then try to classify a potential transaction into a certain sub group by comparing the distances (measured by quantitative attributes) from the potential transaction and the sub groups. In this way, original StereoTrust is improved in the context of online auction by building and combining (quantitative features associated) stereotypes in a more reasonable way. We begin by identifying useful attributes for stereotype formation in the next sub section.

6.3.1 Identifying Attributes

By studying Allegro dataset, we identify four attributes of the transactions:

A_1 : Category of the item.

A_2 : Price of the item.

A_3 : System age of the seller when the transaction occurs.

A_4 : Number of items already sold by the seller when the transaction occurs.

Next we justify the identified attributes using Allegro data. In order to evaluate discrimination power of each attribute, we compute the ratio δ between average value of this attribute for successful transactions and unsuccessful transactions. We only consider the attributes with δ (or $1/\delta$) at least 2 for next steps of our approach¹. Table 6.1 summarizes the ratio δ (or $1/\delta$) for all the attributes. We do not show results of the attribute category (A_1) because its value cannot be computed. Note that the unit for A_3 is hour. According to value of δ (or $1/\delta$), we choose attribute A_2 and A_4 for stereotypes

---

¹Such a criterion works well in the current context. However, it may not be suitable for all other scenarios. It is thus possible to apply other more sophisticated approaches/criteria such as information gain to choose the most discriminating attribute. We leave such an improvement for future work.

creation. It has been reported in the literatures [140, 141] that some dishonest sellers may sell some cheap items in a certain category to gain reputation and then cheat in an auction with high price in another category. So we also consider attribute A_1 when forming the stereotypes.

Table 6.1: Choosing Attributes

Attributes	Avg.(successful)	Avg.(unsuccessful)	δ (or $1/\delta$)	result
A_2	52.59	120.80	2.30	considered
A_3	14713.32	13019.53	1.13	not considered
A_4	358.21	150.00	2.39	considered

6.3.2 Creating Stereotypes Using Identified Attributes

After identifying attributes, we now discuss how to use such attributes and past transactions to create stereotypes. We begin with the attribute of category. A buyer first collects the past transactions. According to the category of the item traded in the transaction, these past transactions are classified into different groups $\{G_{c_1}, G_{c_2}, G_{c_3}, \dots\}$, where c_i indicates category C_i . For group G_{c_i} , we denote the numbers of successful and unsuccessful transactions by N_i^s and N_i^u respectively. The buyer's stereotypes on group G_{c_i} is calculated as $\frac{N_i^s}{N_i^s + N_i^u}$. So when next time this buyer encounters a potential transaction which sells item in category C_i , he/she can estimate trust of this transaction based on its stereotype on group G_{c_i} .

Category based stereotyping works under the assumption that dishonest sellers cheat the buyers in certain categories while acting honestly in other different categories so that buyers finally have clearly different stereotypes on these categories. However, in real world online auction sites, different dishonest sellers may cheat in quite different categories so there is no clear differences among the categories, thus category based stereotyping cannot accurately predict the potential transaction.

We then take into account other attributes which are identified in section 6.3.1. The main idea is to group past transactions on a finer level. For a category based group G_{c_i} , the transactions in this group are further divided into two disjoint sub groups:

a successful sub group $G_{c_i}^s$ and an unsuccessful sub group $G_{c_i}^u$. The buyer assigns a transaction in the group to either sub group according to outcome of this transaction (i.e., if a transaction is successful, it is added to the successful sub group, otherwise, it is added to the unsuccessful sub group). The buyer's stereotypes on successful and unsuccessful sub groups are 1 and 0 respectively².

Each transaction in the sub group has two attributes: price of the item (attribute A_2) and number of items already sold by the seller when the transaction occurs (attribute A_4). So each transaction is represented as a point in a two dimension space, described by its values of attributes A_2 and A_4 . Fig. 6.1 depicts the sub groups of group G_{c_i} in a two dimension space. White points represent successful transactions and black points represent unsuccessful ones. When the buyer encounters a potential transaction, he/she first classifies this transaction into corresponding category formed group and then tries to connect it to successful sub group and unsuccessful sub group based on its values of attributes A_2 and A_4 . Before presenting how to derive trust of the potential transaction using the corresponding stereotypes in Section 6.3.4, we first introduce two kinds of knowledge to form the groups/stereotypes in the next sub section.

Figure 6.1: Sub groups of group G_{c_i} and distances between potential transaction and the two sub groups.

²We assume trust/stereotype value in this work is in $[0,1]$, where 1 represents trustworthy and 0 represents untrustworthy

6.3.3 Local Stereotypes versus Community Stereotypes

To build stereotypes, the past transactions that the buyer knows are classified into different groups. Generally, the buyers' local knowledge reflects their personalized perspective on the system, thus is the most accurate information to create stereotypes (we call them *local stereotypes*). However, in online auction sites, most of the buyers are inexperienced (local knowledge is insufficient). To create stereotypes, inexperienced buyer may use other buyers' past transactions (i.e., community³ knowledge). We rely on ProtoTrust [138] to crawl the auction sites to obtain other buyers' local knowledge. When ProtoTrust plug-in is activated, its crawler works as daemon to collect community knowledge. The crawler gathers information automatically by crawling parts of the auction site (the buyers' comments, transaction archives, the users' profiles). Community knowledge is rather unalterable so this computation can be performed occasionally. The process of gathering community knowledge is stopped after a fixed amount of time or when certain amount of data is obtained. We call the stereotypes created using community knowledge *community stereotypes*. Please note that community knowledge collection is different from the local knowledge sharing overlay network (LKSON) presented in Chapter 4.5 because LKSON, which is particularly designed for decentralized environments, relies on social network information to ensure reliable knowledge sharing, while ProtoTrust works in the centralized environments where social relationships with respect to providing local knowledge among the buyers are absent (coordinated by centralized entity)⁴.

Community stereotypes may not be as accurate as the buyers' local stereotypes, however, they provide the whole community's perspective on the system, which may reveal some information that is unknown to the individual buyers, and are particularly useful for the inexperienced buyers who do not have sufficient local knowledge.

---

³Community refers to a group of users that share the similar properties, e.g., active under certain categories, etc.

⁴Next version of ProtoTrust plans to incorporate such a social network (for decentralized systems) but development of ProtoTrust itself is out of the scope of this thesis (by 3rd party).

6.3.4 Estimating Trust of Transaction Using Stereotypes

After creating stereotypes, we present how to estimate trust of a potential transaction using these stereotypes. We denote the potential transaction by $T(A_2^t, A_4^t)$, where A_2^t and A_4^t denote values of attributes A_2 and A_4 ; and t indicates that these values are associated with transaction T . Similarly, the past transactions (local knowledge or community knowledge) are denoted by $\mathcal{T} = \{T_1(A_2^{t_1}, A_4^{t_1}), T_2(A_2^{t_2}, A_4^{t_2}), T_3(A_2^{t_3}, A_4^{t_3}), \dots\}$. When encountering $T(A_2^t, A_4^t)$, we assume that the groups are already formed: $\mathcal{G} = \{G_{c_1}, G_{c_2}, G_{c_3}, \dots\}$. For group G_{c_i} , its member transactions are divided into two sub groups: successful sub group $G_{c_i}^s$ and unsuccessful sub group $G_{c_i}^u$.

To measure trustworthiness of the potential transaction, the buyer first calculates distances between $T(A_2^t, A_4^t)$ and the two sub groups for each group (i.e., distances between potential transaction and centroids of the two sub groups). The centroids of the two sub groups $(Avg_{G_{c_i}^s}^{A_2}, Avg_{G_{c_i}^s}^{A_4})$ and $(Avg_{G_{c_i}^u}^{A_2}, Avg_{G_{c_i}^u}^{A_4})$ are calculated⁵ as the average values of attributes A_2 and A_4 . Then the distances are calculated as:

$$D_i^s = \sqrt{(Avg_{G_{c_i}^s}^{A_2} - A_2^t)^2 + (Avg_{G_{c_i}^s}^{A_4} - A_4^t)^2} \quad (\text{Eq. 6.1})$$

$$D_i^u = \sqrt{(Avg_{G_{c_i}^u}^{A_2} - A_2^t)^2 + (Avg_{G_{c_i}^u}^{A_4} - A_4^t)^2} \quad (\text{Eq. 6.2})$$

Fig. 6.1 shows distances between potential transaction T and the two sub groups in group G_{c_i} . The stars represent centroids of these two sub groups. The distances demonstrate how close is the potential transaction and two sub groups. That is, if $D_i^s < D_i^u$, potential transaction is closer to successful sub group, which means it is more likely to be successful. Otherwise, this transaction is more likely to be unsuccessful. Such distances can be used to measure weights of the buyer's stereotypes on the two sub groups:

$$w_i^s = \frac{1/D_i^s}{1/D_i^s + 1/D_i^u} \quad \text{and} \quad w_i^u = 1 - w_i^s \quad (\text{Eq. 6.3})$$

---

⁵When calculating the average values, in order to avoid abnormal values, it is necessary to normalize the values to a certain range, e.g., $[0,1]$

It is worth mentioning that if D_i^s or D_i^u is equal to 0, 1 is assigned to w_i^s or w_i^u (and accordingly 0 is assigned to w_i^u or w_i^s). The buyer's stereotypes on successful and unsuccessful sub groups are 1 and 0 respectively. For the potential transaction $T(A_2^t, A_4^t)$, the buyer's stereotype on group G_{c_i} is computed as the weighted sum of its stereotypes on the two sub groups, where weights are measured by the distances⁶:

$$S_i = w_i^s * 1 + w_i^u * 0 \quad (\text{Eq. 6.4})$$

After calculating the buyer's stereotypes on each group, we need to estimate weights of these groups to combine the corresponding stereotypes to derive trust of the potential transaction. We define importance factor λ to indicate the importance of each category/group. Two attributes may influence importance of a category: the number of items sold and prices of the items in this category. If the amount of items sold in a category is large, it means this category is very popular for the buyers thus it deserves more importance. And if the prices of the items in a category are very high, it means the buyers suffer high risks when buying these items thus this category also deserves more importance. So the importance factor λ_i for group G_{c_i} is calculated as:

$$\lambda_i = \sum_j^{N_i} p_j \quad (\text{Eq. 6.5})$$

Where N_i is the amount of items sold in this category and p_j is the price of the j th item. Using important factors, we calculate weight of each group G_{c_i} :

$$W_i = \frac{\lambda_i}{\sum_k \lambda_k} \quad (\text{Eq. 6.6})$$

Where k indicates index of each group. After calculating stereotypes on the groups and their weights, we combine these stereotypes to derive trust of the potential transaction:

$$Tr = \sum_k W_k * S_k \quad (\text{Eq. 6.7})$$

Fig. 6.2 depicts the whole process of combining all stereotypes.

---

⁶Other kinds of weights may also be applied. For instance, if $D_i^s < D_i^u$, we simply set the stereotype on the group as 1 (i.e., the potential transaction clearly belongs to successful sub group).

Figure 6.2: Deriving trust of potential transaction by combining corresponding stereotypes.

Since values of stereotypes are in the range of $[0,1]$, the derived trust value of potential transaction is also in the range of $[0,1]$. Our approach sets the threshold ($\in [0,1]$) to make the binary decision: if derived trust value is larger than the threshold, the potential transaction is considered a successful transaction, otherwise, it is considered an unsuccessful transaction. Such binary decision is used to recommend the buyer to enter or leave the potential transaction⁷. In historical data driven experiments, we check whether the recommendation is appropriate based on whether the actual transaction is successful or not.

6.4 Evaluation

In this section, we conduct experiments over Allegro dataset, as well as synthetic datasets to evaluate performance of our stereotypes based approaches. We first discuss methodology in Section 6.4.1 and then present results in Section 6.4.2 where we first validate performance of StereoTrust based approach and MetaTrust based approach and then compare with other existing representative trust mechanisms.

⁷In any cases, the buyer may choose whatever he/she wishes.

6.4.1 Methodology

The dataset used in this chapter was provided by Allegro, a leading auction site in Eastern Europe⁸. At the beginning of the fourth quarter in 2006, 10,000 sellers and 10,000 buyers have been randomly selected; their profiles and received comments have been stored. During the next 6 months, all transactions conducted by the selected users were monitored and recorded. For every partner who appeared in a transaction and was not in the primary database, all historical information about the received feedback has been collected, but with respect to new auctions only the originally selected users have been monitored. In the first quarter in 2007 the database contained more than 200,000 transactions and over 1.7 million comments (1,712,886 positive feedbacks, 9,437 neutral feedbacks and 5,722 negative feedbacks). A transaction is considered successful if its feedback is positive, otherwise, it is considered unsuccessful⁹. Each transaction is described by three attributes, which are identified in Section 6.3.1: category of the item, price of the item and number of items already sold by the seller when the transaction occurs.

Allegro is a very friendly environment: only around 0.9% of the transactions are labeled negative explicitly. In more generic scenarios, noticeable fractions of agents may behave maliciously (and hence considerably damage the whole system). To test whether our approaches could provide effective decision making guidance for honest users in a hostile environment, and also to compare proposed algorithms with other existing trust mechanisms, we generate synthetic dataset consisting of 10,000 nodes. Each node is either *good*, which provides service satisfactorily or gives genuine feedbacks when requested, or *malicious*, which cheats others in transactions or provides false feedbacks when requested. To make the simulation environment more realistic, there are no completely *good* or *bad* nodes. We set *good* nodes and *malicious* nodes behave maliciously with probabilities of 15% and 85% respectively. We denote the fraction of malicious

---

⁸The data was obtained by us through Dr. Adam Wierzbicki of Polish Japanese Institute of Information Technology

⁹Since most feedbacks are positive, we consider transactions with neutral or negative feedbacks as unsuccessful transactions

nodes in the system by P_m . Any node (trustor) may request service from other nodes (the service provider/the target node). Once the transaction with the service provider (target node) is done, trustor rates the transactions to indicate whether they are successful or not. The trustor stores information of these transactions such as their outcomes and the associated features (see Table 4.1 in Chapter 4.2.3).

We also need to generate synthetic features. Note that each transaction is described by four features¹⁰. Let X_s and X_u be the random variables which represent values of features of successful and unsuccessful transactions respectively. We assume that X_s and X_u follow normal distribution, i.e., $X_s \sim \mathcal{N}(\mu, \sigma)$ and $X_u \sim \mathcal{N}(\mu + \theta', \sigma)$. In our experiments $\mu = 1$, $\sigma = 0.1$ and $\theta' \in [0, 1]$ is the separability factor that we use to tune the overlap between values of features of successful transactions and that of unsuccessful ones. When $\theta' = 0$, values of features for successful and unsuccessful transactions overlap completely, while when θ' is large enough, it means values of features for successful and unsuccessful transactions are separated clearly. In other words, θ' allows us to modify the learnability of the transaction classes. Note that since only our approaches are based on features, θ' only impacts performance of our approaches.

We compare our approaches with two existing trust mechanisms: (1) *Random Selection*: The simplest approach is to engage or reject a potential transaction randomly. We set the probability that the trustor accepts a potential transaction to 50%. This model is used as the experiments benchmark. (2) *Feedback Aggregation*: In this model [19, 26, 80], if trustor does not know target node, it asks other nodes across the network and aggregate feedbacks to derive target node's trust. False feedback filtering mechanism (from TRAVOS [80]) is applied to improve performance.

Note that we only compare our approaches with the existing models when synthetic data is used because we lack essential information to employ all the other models on the Allegro dataset.

The metrics we use to evaluate performance of trust mechanisms include:

---

¹⁰We also conducted experiments using other numbers of features and obtained the similar qualitative trends.

- *false positive rate*

The transaction is unsuccessful but the algorithm predicted that it would be successful.

- *false negative rate*

The transaction is successful but the algorithm predicted that it would be risky.

- *overall falseness rate*

Weighted sum of false positive rate and false negative rate:

$$F_{overall} = \theta * FP + (1 - \theta) * FN \quad (\text{Eq. 6.8})$$

Where $\theta \in [0, 1]$ is harmfulness factor, indicating harmfulness of a certain kind of falseness¹¹. $F_{overall}$ represents overall falseness rate, FP and FN represent rates of false positive and false negative respectively.

6.4.2 Results

6.4.2.1 StereoTrust based approach

False Positive versus False Negative To evaluate performance of our approach that uses individual buyer's local knowledge to create stereotypes, we randomly select 100 buyers with sufficient local knowledge (i.e., at least 15 historical transactions). Then we randomly select 1000 transactions (50% are successful and 50% are unsuccessful). Each buyer evaluates 100 transactions to decide whether to enter the transactions or not. To evaluate our approach that uses community knowledge, we select 10000 buyers at random to form the knowledge pool (community knowledge). Then we evaluate the 1000 randomly selected transactions (50% are successful and 50% are unsuccessful) using community stereotypes. We estimate accuracy of prediction by comparing the decisions made by our approach with the ground truth (real outcomes of these transactions).

Fig. 6.3 demonstrates false positive and false negative caused by StereoTrust based approach that uses the buyer's local knowledge and community knowledge to create

---

¹¹Generally, false positive is more harmful to the buyers than false negative: one can tolerate missing a successful transaction but cannot tolerate losing money in an unsuccessful transaction.

6.3.a: Using the buyer's local knowledge to create stereotypes. 6.3.b: Using community knowledge to create stereotypes.

Figure 6.3: False positive rate versus false negative rate (StereoTrust based approach).

stereotypes respectively. As discussed before, the derived trust value of the potential transactions is in the range of $[0,1]$. To help the buyer make the binary decision, we set the threshold that is also in the range of $[0,1]$. We vary the threshold from 0 to 1 with 0.02 as the increment to observe the changes of false positive rate and false negative rate. From Fig. 6.3.a and 6.3.b, we can see, no matter what kind of knowledge is used, the falseness rates vary with the varying threshold. When threshold increases, false positive rate declines and false negative rate ascends. This is because when threshold is high, the probability that an unsuccessful transaction is predicted as a successful one is low but the probability that a successful transaction is predicted as an unsuccessful one is high (i.e., the transactions whose trust values are not *quite* high are considered unsuccessful). Since false positive rate and false negative rate change with thresholds conversely, it is difficult to set a threshold to keep rates of the two kinds of falseness lowest simultaneously.

Overall Falseness We define the overall falseness to combine false positive and false negative by taking into account harmfulness factor θ (see Eq. 6.8). Fig. 6.4 shows the overall falseness of our approach that uses the buyer's local knowledge and community knowledge to create stereotypes respectively. We vary harmfulness factor θ from 0.6 to 0.9 with 0.1 as increment. From Fig. 6.4.a and Fig. 6.4.b, we observe that the general

6.4.a: Using the buyer's local knowledge to create stereotypes. 6.4.b: Using community knowledge to create stereotypes.

Figure 6.4: Overall falseness (StereoTrust based approach).

trend of overall falseness rate is: it declines with the increasing threshold and stops at a certain point, then it starts ascending. So the lowest point is the optimal result that our approach is able to achieve and the corresponding threshold is the design parameter we are looking for. From the two figures, we also observe that no matter which kind of knowledge is used, when θ is bigger, the corresponding overall falseness rate is lower. This is because false negative rate is higher than false positive rate (i.e., try to keep false positive as few as possible). So when $(1 - \theta)$ becomes smaller, $(1 - \theta) * FN$ (see Eq. 6.8) makes the overall falseness rate lower despite the fact that $\theta * FP$ becomes bigger.

Local Stereotypes Versus Community Stereotypes Now we compare performance of local stereotypes and community stereotypes. Fig. 6.5 shows the rate of overall falseness caused by our approach that uses local stereotypes and community stereotypes respectively. We set the harmfulness θ as 0.8 and 0.9 respectively. From the figure we observe that under the same θ , local stereotypes based approach outperforms community stereotypes based approach. When $\theta = 0.8$, the lowest rate of overall falseness for local stereotypes based approach is 0.029 and that for community stereotypes based approach is 0.0626. When $\theta = 0.9$, the lowest overall falseness rate for local stereotypes based approach is 0.027 and that for community stereotypes based approach is 0.038. This is because local stereotypes are derived using individual buyer's

Figure 6.5: Comparison of performance of local stereotypes and community stereotypes (StereoTrust based approach).

Table 6.2: Summary of overall falseness rates and the corresponding thresholds (StereoTrust based approach)

	Local (% of overall falseness)	Local (threshold)	Community (% of overall falseness)	Community (threshold)
$\theta = 0.60$	3.30%	0.60	10.86%	0.84
$\theta = 0.70$	3.10%	0.60	8.57%	0.84
$\theta = 0.80$	2.90%	0.60	6.26%	0.86
$\theta = 0.90$	2.70%	0.60	3.80%	0.86

local knowledge, which is the most accurate information source for that buyer while community stereotypes take into account other buyers' opinions which may be different from this buyer's opinion. However, we notice that even if local stereotypes are more accurate than community stereotypes, the difference is not very big. Moreover, community stereotypes are particularly useful for the inexperienced buyers, which almost dominate the online auction sites. So we believe community stereotypes are also useful and can be seen as a kind of complementary knowledge. Table 6.2 summarizes the overall falseness rate and the corresponding thresholds for local knowledge based approach and community knowledge based approach with different θ values.

6.4.2.2 MetaTrust based approach

We next evaluate MetaTrust based approach in a slightly different way. We first sort the 10,000 buyers according to the number of their past transactions, i.e., the first buyer has the most past transactions and the last one has the least transactions. We then select subset U_b of these buyers starting from the first one. Each buyer evaluates 100 randomly selected transactions (50% are successful and 50% are unsuccessful). We vary the size of U_b to investigate effect of local knowledge volume.

Fig. 6.6.a demonstrates performance of MetaTrust based approach (LDA and DT based), i.e., how average rates of false positive and false negative evolve when U_b varies from 5 to 500. As expected, all falseness rates increase when U_b grows. This shows the impact of local knowledge on our proposed algorithms: when U_b is small, it contains only experienced buyers who have enough past transactions to allow MetaTrust based approach to issue accurate predictions. As U_b grows, it contains more and more inexperienced buyers, for which the predictions are less accurate. We also notice that DT based algorithm is less accurate than LDA based algorithm. This is because basic decision tree is more suitable for qualitative features while features identified in auction data are quantitative (i.e., it is not easy to obtain the appropriate thresholds to efficiently split the training data.).

Fig. 6.6.b shows the distribution of numbers of individual buyers' past transactions (only first 3000 are shown). Note the logarithmic scale for y-axis: the number of past transactions is quickly decreasing. For instance, less than 100 buyers have more than 20 past transactions. Estimating the minimal number of transactions that allow MetaTrust based approach to be precise is challenging, since not all transactions have the same importance (e.g., two very different transactions will be much more useful for MetaTrust based approach than two identical transactions). However, in this set of experiments, we estimate empirically that when numbers of transactions is over 6, the potential transaction can be relatively reliably predicted (i.e., the overall falseness rate is smaller than 0.1 for LDA based algorithm and 0.19 for DT based algorithm). Simulation results thus

6.6.a: Performance of MetaTrust with varying num-6.6.b: Numbers of individual buyers' past transaction.

Figure 6.6: Experiments using Allegro dataset (MetaTrust based approach).

show that MetaTrust based approach is capable of quickly “learning” what a successful or unsuccessful transaction is.

In online auction sites, most of the buyers are inexperienced (local knowledge is insufficient). So to predict trustworthiness of the potential transaction, the inexperienced buyer will collect other buyers' local knowledge to perform MetaTrust based approach by constructing LKSON. LKSON is initialized by connecting this buyer and other buyers according to categories that they are interested in (i.e., the item this buyer is going to buy is in the category from which the connected buyers used to buy items). Third party knowledge indeed helps inexperienced buyer predict trustworthiness of the potential transaction, however, it also introduced inaccurate information, which may influence accuracy of prediction because third party knowledge only reflects knowledge provider's personalized perspective on the system. For instance, one buyer can tolerate one day delay of item delivery (successful transaction) while another buyer cannot (unsuccessful transaction). We thus need to estimate how many other buyers are requested to gather sufficient knowledge while trying to keep influence of inaccurate information as low as possible.

Fig. 6.7 demonstrates how performance of MetaTrust based approach (using LDA) varies with the varying number of requested buyers (from 10 to 110 with 10 as increment). We observe that the general trends of the false positive rate and false negative

rate are: they decline with the increasing number of requested buyers and stop at a certain point, then they start ascending. This is because when the number of requested buyers is low, there are not sufficient past transactions available thus affecting the accuracy of prediction (i.e., not statistically significant). On the other hand, when the number of requested buyers is high, lots of transactions are collected, which means more inaccurate information is incorporated, hence the accuracy of prediction is also influenced. So the lowest point is the optimal result that MetaTrust based approach is able to achieve and the corresponding number of requested buyers is the design parameter we are looking for. From the figure we can see the most suitable number of requested buyers in this scenario is 60 and the corresponding false positive rate and false negative rate are 0.08 and 0.042 respectively.

Figure 6.7: Performance of MetaTrust based approach (LDA) with LKSON.

Comparison We compare performance of MetaTrust and StereoTrust based approaches with other existing trust models (Fig. 6.8, 6.9, 6.10 and 6.11). Note that for simplicity, the overall falseness rate is simply the sum of false positive rate and false negative rate without any weights. We observe that when percentage of malicious nodes P_m is low, feedback aggregation model performs quite well. This is because when most of nodes in the system are honest, the aggregated feedbacks can accurately predict behavior of the service provider thus keeping rates of false positive, false negative as well as overall

6.8.a: False positive.

6.8.b: False negative.

6.8.c: Overall falseness.

 Figure 6.8: Performance comparison using synthetic dataset ($\theta' = 0.2$).

6.9.a: False positive.

6.9.b: False negative.

6.9.c: Overall falseness.

 Figure 6.9: Performance comparison using synthetic dataset ($\theta' = 0.4$).

falseness low. However, with the increase of P_m , various falseness increases sharply. So accuracy of feedback aggregation model heavily depends on percentage of malicious nodes and thus is not suitable for a hostile environment.

One interesting thing about feedback aggregation model is that its false negative rate decreases when P_m is larger than 0.5 (Fig. 6.8.b, 6.9.b, 6.10.b and 6.11.b). The possible reason is that when P_m is high, the amount of false feedbacks increases thus increasing the falseness probability. On the other hand, the total amount of successful transactions decreases generally, so the false negative rate decreases.

Random selection model, as we expected, provides a stable rate of overall falseness of 0.5 because we let trustor accept or reject transactions equiprobably. This model is the benchmark of the experiments, demonstrating falseness rate evolution in the scenario

6.10.a: False positive.

6.10.b: False negative.

6.10.c: Overall falseness.

 Figure 6.10: Performance comparison using synthetic dataset ($\theta' = 0.6$).

6.11.a: False positive.

6.11.b: False negative.

6.11.c: Overall falseness.

 Figure 6.11: Performance comparison using synthetic dataset ($\theta' = 0.8$).

where no trust management is applied.

StereoTrust based approach outperforms feedback aggregation in general especially when P_m is high. This is because StereoTrust based approach mainly relies on a trustor's local knowledge to form (local) stereotypes. Only when local knowledge is not available, it collects community knowledge to form (community) stereotypes. Since ProtoTrust does not provide the function of filtering out biased information, StereoTrust based approach may also suffer from false third party information, but such a probability is much lower than feedback aggregation, which mainly relies on feedbacks.

For MetaTrust based approach, we observe that DT based approach is less accurate than LDA based approach in all scenarios. This result conforms to the result when real data is used (Fig. 6.6.a) and further proves that LDA based algorithm is more suitable to the scenarios where features are quantitative. So to sum up, LDA based approach, which has the lowest rate of various falseness outperforms all other models (except for the scenarios that θ' is quite small, i.e., when $\theta' \leq 0.4$). When θ' is quite small, both LDA based approach and DT based approach suffer from low accuracy of classification. Finding features that can distinguish successful transactions from unsuccessful ones is thus the key of MetaTrust based approach. We also notice that performance of MetaTrust based approach is quite stable (not affected by P_m very much). This is because MetaTrust based approach mainly relies on local knowledge, which is the most reliable information source. Note that since MetaTrust based approach relies on different kinds of information, it may not be rather fair to compare with other models that only use one kind of information (i.e., the target seller's historical information). Moreover, in some application scenarios, the information used by MetaTrust based approach (e.g.,

features) may not be always available. Nevertheless, whenever such information is there, MetaTrust based approach indeed produces reasonable results thus demonstrating its efficiency.

6.5 Conclusion

In this chapter, we apply MetaTrust and StereoTrust in the context of online auction to validate their practicality (i.e., detecting fraud auctions). We conducted comprehensive experiments over a large-scale real auction dataset to evaluate performance of both StereoTrust based approach and MetaTrust based approach. Simulation results show that for StereoTrust based approach, local stereotypes based trust is more accurate than community stereotypes based trust, but the advantage is not quite obvious. We also find out the thresholds that achieve the lowest overall falseness rate. That are 0.6 for local stereotypes based approach and 0.84/0.86 for community stereotypes based approach. These values can make suggestions to the system designers on parameter configuration. For MetaTrust based approach, not surprisingly, its performance is positively correlated with the discrimination power of features on successful interactions and unsuccessful interactions. However, compared to other approaches, MetaTrust based approach (particularly, LDA) is quite efficient, especially when third party information is not reliable. Moreover, its performance is quite stable because (1) it mainly relies on a trustor's local knowledge and (2) its LKSON (see Chapter 4.5) provides reliable third party information sharing.

While the approaches discussed so far look at an agent's trust using mechanisms beyond reputation, they do so under the premise that a specific agent's behavior is independent of the time/life-cycle of its presence in the system. In practice, agents' behavior may change or fluctuate. Next we study mechanisms to predict agent behavior under such dynamism.

Publication(s)

- *Using Stereotypes to Identify Risky Transactions in Internet Auctions*,
Liu Xin, Tomasz Kaszuba, Radoslaw Nielek, Anwitaman Datta, Adam Wierzbicki
Symposium on Social Intelligence and Networking (SIN), in Proceedings of the 2nd
IEEE International Conference on Social Computing (SocialCom), Minneapolis,
Minnesota, USA, 2010.

Part IV

Trust capturing dynamic behavior

In the previous chapters, we present the theories of StereoTrust and MetaTrust, and their applications in diverse distributed environments. These works demonstrate our attempts in solving bootstrapping issue in trust and reputation systems. We then go back the (more common) scenario where the target agent’s historical information is available (i.e., the second drawback of current reputation based trust mechanisms identified in introduction section). Traditional reputation based trust mechanisms focus on extracting accurate knowledge from various information sources and utilize the same to assess trust by investigating outcomes of the target agent’s past transactions. Such a method, while effective in distinguishing malicious agents from the permanently well-behaved ones, may not efficiently detect the dynamic behavior of the intelligent agent who may strategically change his behavior to maximize the profits. In this part, we present two approaches (beyond reputation), looking at two aspects of the problem. In chapter 7, we propose a trust prediction approach capturing an agent’s dynamic behavior. We study the long term past behavior of the target agent to identify relevant patterns for predicting trustworthiness of the potential transaction. Differently, in Chapter 8, we particularly look at the specific stages of the target agent’s life cycle. That is, we try to detect the imprudence behavior of the target agent when he is already considered ‘reliable’ by existing reputation systems.

Please note that given different focuses, it is unfair to compare the two solutions. For instance, the solution presented in Chapter 8 only works effectively when the target agent’s reputation value is high, while the solution presented in Chapter 7 works without restrictions of reputation value, but relies on some specific assumptions.

Chapter 7

Detecting agents' dynamic behavior based on context matching

Existing reputation based trust mechanisms derive trust by statistically investigating the target agent's historical information. Most of these approaches work under the premise that a specific agent's behavior is independent of the time/life-cycle of its presence in the system (i.e., a static behavior model¹). However, in practice, an intelligent agent may strategically change his behavior to maximize the profits. In this chapter, we therefore propose a trust prediction approach aiming at capturing dynamic behavior of the target agent [142]. Specifically, we first identify features which are capable of describing/representing context of a transaction. Then we use these features to measure similarity between context of the potential transaction and that of previous transactions to estimate trustworthiness of the potential transaction based on previous similar transactions' outcomes. Evaluation using real auction data and synthetic data demonstrates efficacy of our approach in comparison with an existing representative trust mechanism.

This chapter is organized as follows: In section 7.1, we describe the background and motivation of this work, as well as outline our approach. In section 7.2, we elaborate a basic trust prediction approach, propose further refinements and analyze its computational complexity. Evaluation using real auction dataset and synthetic dataset

---

¹Some works apply fresh factor/forgetting rate to pay more attention to the recent transactions to deal with possible changes of an agent's behavior over time [19, 26, 33, 20]. However, such an approach ignores an agent's older past behavior patterns, which may also be valuable for future behavior prediction. Differently, as described later, our approach studies an agent's long term past behavior to try to detect the most appropriate patterns for guiding trust assessment.

is conducted to quantify the performance of the proposed approach which we compare with a representative trust mechanism in Section 7.3. We finally summarize this work with discussion of possible future directions in Section 7.4.

7.1 Introduction

In the context of reputation based trust assessment, various techniques (e.g., Bayesian approach [80, 20, 44], eigen matrix [25], subjective logic [88], etc.) are applied to study the target agent's past behavior. However, these works assume a relatively static agent behavior and are thus not able to capture an agent's dynamic behavior patterns, which are common in a large scale open system since self-interested agents are very likely to vary their behavior to achieve their goals. For instance, in an online auction site, a malicious seller may act honestly in selling cheap items to gather sufficient reputation and then cheat in selling an expensive item. Traditional trust mechanisms, even though give more weights to the recent transactions to deal with possible behavior changes, are likely to incorrectly predict the risky transaction as safe since a majority of the seller's recent transactions were good.

In this work, we propose a trust prediction approach based on the target agent's historical information, aiming at capturing his dynamic behavior pattern to accurately estimate trustworthiness of the potential transaction with this agent. The main idea is that outcome of the potential transaction can be indicated by that of certain previous transactions, which have the most similar transaction context with the potential transaction. Specifically, given a set of a trustor's past transactions (in chronological order) with the target agent (i.e., direct experience), we first define context of a transaction as a sequence of its immediate previous transactions (a transaction window), each of which is described by a feature vector. Note that feature selection is application dependent and we will demonstrate in evaluation section which features of a transaction are identified using online auction as an example. We then compare similarity of the context of the potential transaction and that of previous transactions. The outcome of

a specific previous transaction that has the most similar context is used as the indicator to estimate trustworthiness of the potential transaction. The similarity between the two contexts (with the same window size) is calculated by aggregating feature based similarity between a pair of previous transactions in the corresponding contexts.

In case that trustor's direct experience is not available/sufficient, he will contact other agents for indirect experience. The direct experience and indirect experience are merged (in chronological order) to serve as the knowledge repository (i.e., training data) for trust prediction. Note that in this work, we do not discuss how to filter out false indirect experience, which has been thoroughly studied in the community [80, 27, 35, 20]. In order to further improve prediction accuracy, we propose to use multiple transaction window sizes and apply dirichlet distribution to model multiple trust indicators.

7.2 Our approach

7.2.1 Notation

We denote by $\mathcal{A}$ the set of all agents in the system. We assume two types of agents, *customers*, which request service from other agents, i.e., *providers*, which provide service. A transaction happens when a customer accepts a provider's service. To indicate quality of a service, the customer can rate the transaction, where rating is a discrete quantitative variable in a certain range, denoted by $\mathcal{L} = \{L_1, L_2, \dots, L_l\}$. For instance, the rating could be in the range of [1, 2, 3, 4, 5], where 1 to 5 represents *lowest*, *low*, *medium*, *high* or *highest* respectively. Θ_{a_x, a_y} denotes the transaction between provider agent a_y and customer agent a_x . Each transaction Θ is associated with a feature vector $F_\Theta = \{f_\Theta^1, f_\Theta^2, \dots, f_\Theta^d\}$. Note that in order to efficiently combine various features, we normalize values (for each feature) to the same range, i.e., [0,1]. The features can be obtained by agent's profile or context of the transaction. For instance, in an online auction site, the features can be item's category, price or period of the auction, etc. We will detail such an example in evaluation section.

We denote consumer a_x 's past transactions with the provider a_y by $\mathcal{T}_{x,y}^D = \{\Theta_{a_x,a_y}^1, \Theta_{a_x,a_y}^2, \dots\}$, where D indicates direct experience. The indirect experience, i.e., past transactions obtained from other agents who have interacted with a_y are denoted by $\mathcal{T}_{x,y}^I$. Given a set of a_y 's past transactions (ordered chronologically), we can study the pattern of a_y 's behavior by setting transaction context (i.e., transaction window), which is actually sequential subset of the transaction set, denoted by W_s , where s represents size of this transaction window.

7.2.2 Using direct experience

Consider the scenario where customer a_x needs to predict trustworthiness of a potential service provider a_y to decide whether to interact with it or not. When a_x has sufficient past transactions with a_y , a_x can utilize these as the knowledge pool to investigate a_y 's behavior pattern to predict trustworthiness of the potential transaction with a_y .

The main idea is that, given the most recent transaction window W_s^r of a_x 's sequential past transactions $\mathcal{T}_{x,y}^D$ with a_y (see Fig. 7.1 as an example), we select the most *similar* earlier transaction window and estimate trustworthiness of the potential transaction Θ based on rating of the transaction which is immediately following the selected earlier transaction window. The similarity between transaction windows reflects how similar the context/situation of a specific past transaction is to that of the potential transaction Θ . The essential intuition is that the outcome of the past transaction which has the most similar transaction context is a promising indicator of the real quality of Θ .

Algorithm 2 illustrates how our approach predicts trustworthiness of a potential transaction in detail. Given trustor a_x 's sequential past transactions (in chronological order) $\mathcal{T}_{x,y}^D$ with a_y , based on size of the recent transaction window W_s^r , a_x obtains the earliest transaction window W_s^e , which is actually a sequence of the first s transactions in $\mathcal{T}_{x,y}^D$. Based on features of the transactions, a_x calculates similarity between W_s^r and W_s^e using function $getSimilarity(W_s^r, W_s^e)$ (Line 6 in Algo. 2), which will be described in detail in the next sub section. a_x then slides W_s^e to the right by one position to obtain

a new W_s^e and calculates the similarity between W_s^r and the new W_s^e . This process continues until W_s^e reaches the second last past transaction. After similarity calculation, a_x chooses the earlier transaction window which has the largest similarity score and obtains outcome (say $L \in \mathcal{L}$) of the past transaction which is immediately following that transaction window. Then outcome of the potential transaction is estimated as L . a_x finally decides whether to interact with a_y or not based on such prediction. For instance, if the predicted trustworthiness is higher than a predefined threshold, a_x enters the transaction, otherwise, a_x rejects a_y and evaluates other potential interaction partners.

Note that when a_x encounters a new potential transaction (i.e., after current potential transaction), the recent transaction window W_s^r is shifted to the right by one position. So our approach is able to adapt to newly encountered transactions. We next discuss how to calculate similarity between two transaction windows in detail.

Algorithm 2 Trust prediction algorithm

- 1: Given a_x 's sequential past transactions (in chronological order) $\mathcal{T}_x^D$ with a_y , based on size of the recent transaction window W_s^r , a_x obtains the earliest transaction window W_s^e , which is the first s transactions in $\mathcal{T}_x^D$.
  - 2: $S' = 0$ //initial similarity score.
  - 3: $w = -1$ //index of the earlier transaction window
  - 4: **while** TRUE **do**
  - 5: a_x calculates similarity between W_s^r and W_s^e :
  - 6: $S \leftarrow \text{getSimilarity}(W_s^r, W_s^e)$.
  - 7:   **if** $S > S'$ **then**
  - 8: $S' = S$
  - 9: $w = \text{index of current } W_s^e$.
  - 10:   **end if**
  - 11:   **if** current W_s^e reaches last past transaction but one of $\mathcal{T}_x^D$ **then**
  - 12:   Breaking WHILE loop.
  - 13:   **else**
  - 14:   Sliding W_s^e to right by one position to obtain a new W_s^e .
  - 15:   **end if**
  - 16: **end while**
  - 17: a_x obtains outcome (say $L \in \mathcal{L}$) of the past transaction which is immediately following the w th W_s^e .
  - 18: a_x then predicts trustworthiness of the potential transaction Θ is L .
  - 19: a_x finally decides whether to interact with a_x or not based on the prediction.
-

7.2.2.1 Similarity calculation

There exist many methods to measure similarity between a pair of objects (e.g., Jaccard index², cosine similarity³, etc.). Given feature vector of each transaction, we apply a heuristic based on Euclidean distance to calculate similarity between the two transaction windows.

For the i^{th} ($i \in [1, s]$) transaction of the two transaction windows $\Theta_i \in W_s^r$ and $\Theta'_i \in W_s^e$, we calculate similarity between these two transactions using their features (Note that all transactions have the same set of features):

$$\sigma_{\Theta_i, \Theta'_i} = \frac{1}{\sqrt{\sum_{j=1}^{|F_{\Theta}|} (f_{\Theta_i}^j - f_{\Theta'_i}^j)^2}} \quad (\text{Eq. 7.1})$$

$\sigma_{\Theta_i, \Theta'_i}$ represents similarity between the two transaction windows at the i th transaction. Following chronological order, a_x is able to measure the overall similarity of the two transaction windows W_s^r and W_s^e by aggregating similarity of all transaction pairs (see Fig. 7.1 as an illustrative example where size of transaction window is 3):

$$S_{r,e} = \sum_{i=1}^s \sigma_{\Theta_i, \Theta'_i} \quad (\text{Eq. 7.2})$$

By taking into account multiple features, a_x can comprehensively measure how similar is the earlier transaction window to the recent transaction window. Clearly, the more similar the two transaction windows, the outcome of the potential transaction is more accurately predicted based on that of the past transaction which is immediately following that earlier transaction window.

7.2.2.2 Improving prediction accuracy

We notice that if window size varies, the prediction outcome may also vary accordingly since transaction windows with different sizes may demonstrate different transaction context (i.e., may include or ignore certain behavior patterns). Therefore, in order to

---

²http://en.wikipedia.org/wiki/Jaccard_index

³http://en.wikipedia.org/wiki/Cosine_similarity

Figure 7.1: Measuring similarity between recent transaction window and earlier transaction window.

improve prediction accuracy, we vary window size (for both recent transaction window and earlier transaction windows) to obtain multiple prediction results for a single prediction. Now the question is how to handle these results to derive the most accurate final prediction result. Given multiple levels of trustworthiness (see Section 7.2.1 for notations) of a transaction, we model the prediction results using dirichlet distribution [143], which captures a set of observations that have multiple possible outcomes.

We assume for a single prediction, there are W different window sizes and the corresponding prediction results are denoted by $L' = \{L_k | k = 1, \dots, W; L_k \in \mathcal{L}\}$. We then obtain dirichlet probability density function:

$$f(\vec{p}|\vec{\alpha}) = \frac{\Gamma(\sum_{i=1}^k \alpha_i)}{\prod_{i=1}^k \Gamma(\alpha_i)} \prod_{i=1}^k p_i^{\alpha_i-1}, \quad (\text{Eq. 7.3})$$

Where $\vec{p} = \{p_i \geq 0 | 1 \leq i \leq k\}$ denotes the $|\mathcal{L}|$ -component random probability variable, and $\vec{\alpha} = \{\alpha_i | 1 \leq i \leq k\}$ denotes the number of predictions corresponding to one of the possible outcomes (derived from L'). The expected value of any of the $|\mathcal{L}|$ random variable (i.e., expected probability of a certain prediction result) is:

$$E(p_i|\vec{\alpha}) = \frac{\alpha_i}{\sum_{j=1}^k \alpha_j} \quad (\text{Eq. 7.4})$$

An intuitive way to decide the final result from multiple predictions is that the final prediction $L_k \in \mathcal{L}$ is the prediction result which has the highest expected probability.

7.2.3 Incorporating indirect experience

When service consumer a_x does not have sufficient past transactions with the potential service provider a_y (which is common in large-scale, open systems), he may resort to other agents who have interacted with a_y . However, selfish or dishonest agents may refuse to provide their experience with a_y or report false information. Many works [26, 80, 27, 20] have been proposed to address false reports but discussion of such defence mechanisms is not the focus of this work. That is, our work provides a novel method on how to utilize direct experience and indirect experience for dynamic trust assessment. The issue regarding how such information is efficiently collected, although is very important, has been thoroughly studied in the community so we do not spent effort on this issue but just borrow existing mechanisms to ensure that our approach could work properly.

By contacting relevant agents, a_x collects a set of indirect past transactions with a_y , denoted by $\mathcal{T}_{x,y}^I$. Such indirect experience are merged with a_x 's direct experience (if any) to form a knowledge base: $\mathcal{T}_{x,y} = \mathcal{T}_{x,y}^D \cup \mathcal{T}_{x,y}^I$. Utilizing $\mathcal{T}_{x,y}$, a_x is able to perform our proposed trust prediction algorithm (Algo. 2) to predict trustworthiness of the potential transaction. Note that the mixed direct experience and indirect experience are also sorted in chronological order.

7.2.4 Computational complexity

We assume that there are N (direct and/or indirect) past transactions. When single transaction window is considered, we denote the size of transaction window by S ($< N$) (for both recent transaction window and earlier transaction window). According to our algorithm, for a single trust prediction, there are $N - S$ transaction window similarity calculations and in each such calculation, trustor calculates S times of similarity between a pair of past transactions. So the computational complexity of a single trust prediction is $O(NS - S^2)$.

When multiple transaction window sizes are considered for prediction accuracy improvement, the computational complexity changes accordingly. We denote set of possible

windows size by $\mathcal{S}$. Note that $S_i(\in \mathcal{S}) < N$. So the computational complexity becomes $O(N \sum_{i=1}^{|\mathcal{S}|} S_i - \sum_{i=1}^{|\mathcal{S}|} S_i^2)$.

Computational complexity of our approach depends on number of past transactions N , transaction window size S_i and number of window sizes $|\mathcal{S}|$. We can manipulate these three variables to control computational complexity. In order to comprehensively study target agent's past behavior, we do not suggest to reduce N (i.e., removing old transactions) except that the number of past transactions is so large that computation speed is seriously slowed or the past behavior is obviously different from recent ones (e.g., a seller only got one or two negative comments when he just joined the system and then got all positive comments for the following 1000 transactions). Alternatively, we suggest to choose appropriate window size and number of these sizes such that the computational overhead is suitably reduced and trust prediction accuracy is not affected greatly. We will evaluate such discussion in the evaluation section.

7.3 Evaluation

7.3.1 Simulation settings

We use real dataset collected from an Internet auction site Allegro as well as synthetic data to conduct experiments. Please refer to Chapter 6.4 where the same Allegro dataset has been described and used in the previous experiments. We assume binary outcome of a transaction, i.e., a transaction is considered successful if its feedback is positive, otherwise, it is considered unsuccessful. In order to fully understand how a seller changes its behavior in the transactions, we select a set of (150) sellers which have sufficient historical information (i.e., over 100 past transactions).

According to our approach, each transaction is described by a feature vector. By studying Allegro dataset, we identify five features which can be used to describe context of a transaction: (1) category of the item; (2) price of the item; (3) period of the auction (i.e., difference between auction end time and auction start time); (4) number of items already sold by the seller at the time the transaction happens and (5) outcome of the

transaction. Note that in order combine these features, we first normalize their values to the range of $[0,1]$ (for feature (2), (3) and (4)); for item category, if two transactions are within the same category, the difference of this feature is 0, otherwise, the difference is 1 (see Eq. 7.1); for transaction outcome, positive outcome is translated to 1 and negative outcome is translated to 0.

Real allegro dataset provides a real evaluation environment, however, behavior patterns of the sellers in real data are predetermined, and for which we do not have ground truth. In order to comprehensively evaluate performance of our approach under different circumstances, and also to more flexibly control agents' behavior, we generate synthetic data derived from real data. Specifically, we generate a synthetic seller with 100 past transactions and simulate three types of his dynamic behavior: (i) for a transaction, the seller cheats or not randomly; (ii) the seller provides good service for the first half of the past transactions, then followed by bad quality services for the remaining half; (iii) the seller conducts several good transactions and then followed by a bad one (simulating the real scenario where a seller behaves honestly in selling cheap items to aggregate sufficient reputation and then cheat in a transaction selling an expensive item). In order to make the simulation more realistic, we further adjust outcomes of the transactions by configuring that a original good/bad transaction is good/bad with the probability of 0.85, otherwise, it is bad/good. Note that all the transactions features (and values) are taken from real Allegro dataset (i.e., a good/bad synthetic transaction is generated from a randomly selected good/bad real transaction). Using synthetic data, we compare our approach with a personalized trust mechanism⁴ [20], which is closest to our approach (see Chapter 2.2.6 for a brief description). We call this compared approach PTE (short for personalized trustworthiness evaluation). Each experiment is repeated 30 times and error bars are added to indicate deviation across the experiments.

For both real and synthetic dataset, we use false positive rate and false negative rate as the metrics to evaluate the performance of the trust mechanisms.

---

⁴Since we assume a centralized simulation environment, i.e., online auction site, we assume that there is no false indirect experience.

7.3.2 Results

We first demonstrate performance of our approach using real Allegro dataset. Fig. 7.2.a and 7.2.b show false positive rate and false negative rate respectively when transaction window sizes vary from 2 to 20 (a specific window size is applied to all sellers). We observe that different transaction window sizes result in different trust prediction accuracy. The general trends of the falseness rates are: they ascend and then fall to the lowest point, and then ascend again. We show that when window size is 12, we obtain the lowest false positive rate and false negative rate simultaneously. The results indicate that by investigating impact of transaction window size, trustor is able to determine the optimal window size to achieve high trust prediction accuracy while incurring reasonable computational overhead (see Section 7.2.4)

Figure 7.2: Experiments using Allegro dataset (one transaction window size).

We then conduct experiments to demonstrate how trust prediction accuracy is improved by applying multiple transaction window sizes. The transaction window size range $[2,25]$ is split into 5 sub ranges, i.e., $[2,5]$, $[6,10]$, $[11,15]$, $[16,20]$ and $[21,25]$. We show falseness rates by applying different transaction window size sub ranges and compare the results with that when general range $[2,25]$ is applied in Fig. 7.3. Obviously, by applying the general range, potential transaction's context is comprehensively studied thus generating the most accurate trust prediction. However, this incurs high computational overhead. We also observe that by simply applying certain sub ranges (e.g., $[11,15]$, $[16,20]$), high trust prediction accuracy can still be achieved. This again

demonstrates that by choosing appropriate transaction window size (range), our approach provides high trust prediction accuracy while keeping computational overhead reasonable.

Figure 7.3: Experiments using Allegro dataset (multiple transaction window sizes).

We finally compare our approach with PTE using synthetic dataset. The comparison is conducted under three types of target seller's dynamic behavior patterns (see simula-

Figure 7.4: Our approach vs PTE (synthetic dataset, multiple transaction window size).

tion setting in Section 7.3.1). From Fig. 7.4.a we observe that under behavior pattern (i) (i.e., the seller cheats or not randomly), for PTE, both false positive rate and false negative rate are kept at roughly 0.5. This is because in all time windows, trustor counts the similar numbers of successful and unsuccessful past transactions, which makes PTE approximate random selection. While our approach tries to match context of the potential transaction with that of previous transactions to estimate trustworthiness of the potential transaction, which demonstrates better results.

When behavior pattern (ii) is applied (see Fig. 7.4.b)⁵, PTE performs poorly at the beginning because most of past transactions are successful, which make PTE incorrectly predicts that the potential transactions are more likely to be successful (which are actually risky). Later on, When more unsuccessful transactions are encountered, PTE adjusts the model parameters to detect the risky transactions. So the overall false positive rate is around 0.365. On the other hand, although our approach cannot accurately predict potential transaction's trustworthiness at the very beginning either, it is capable of quickly learning transactions' contexts thus lowering the overall false positive rate (around 0.25 when transaction window size range is [16,20]).

In Fig. 7.4.c where behavior pattern (iii) (i.e., the seller behaves well for long, and then suddenly misbehaves for one transaction) is applied, PTE has false positive rate as high as around 0.65. This is because according to this behavior pattern, most of past transactions are successful, which is very likely to generate positive predictions. That is to say, PTE only looks at relative frequency of (un)successful transactions, while our approach is able to discern them since it is based on context rather than just the relative frequencies.

By comparing with PTE, we observe that under different behavior pattern scenarios, our approach outperforms PTE in general by matching transactions' contexts. Moreover, by choosing appropriate transaction window size range (e.g., [16,20] in this case), our approach improves trust prediction accuracy significantly.

---

⁵Since the seller mainly acts honestly for the first half transactions and acts dishonestly for the rest ones, we only show false positive rate.

7.4 Conclusion

In this work, we focus on the impact of agents' dynamic behavior on trust prediction. Our approach relies on agent's past behavior to compare contexts of the past transactions with that of a potential transaction. Trustworthiness of the potential transaction is then estimated based on outcome of a specific past transaction which has the most similar context. The context of a transaction is actually a transaction window just before that transaction and similarity between two transaction windows is calculated based on features of the corresponding transaction pairs.

Evaluation using real auction dataset and synthetic dataset demonstrates that by carefully choosing transaction window size, our approach is capable of accurately predicting trustworthiness of a potential transaction based on an agent's past transactions (i.e., detecting risky transaction). Moreover, in synthetic data based experiments, under multiple behavior patterns, our approach outperforms existing representative trust mechanism.

Currently, the accuracy of this approach is greatly influenced by the sequence of the target agent's past transactions, which, however, does not always hold in reality. The discussed approach is effective when an agent changes his behavior frequently and in a specific pattern, but is not well suited to identify relatively infrequent dishonest behavior of an agent with high reputation, and hence considered 'reliable' by traditional reputation based models. In the next chapter, we look at the more specific stage of an agent's life cycle to predict misbehavior among agents with high reputation. That is, when an agent is already considered 'reliable' by traditional reputation systems, how to detect his inappropriate behavior, e.g., due to imprudence.

Publication(s)

- *A trust prediction approach capturing agents' dynamic behavior* (poster),

Liu Xin, Anwitaman Datta

In Proceedings of the 22nd International Joint Conference on Artificial Intelligence (IJCAI), Barcelona, Spain, 2011.

Chapter 8

Detecting Imprudence of ‘Reliable’ Sellers in Online Auction Sites

In the previous chapter, we described a trust prediction approach focusing on detecting dynamic behavior of an agent based on transaction context matching. Such an approach is particularly effective when an agent repeats his behavior patterns. However, this assumption is not always true in reality. For instance, certain service providers may act honestly for a long time. But when he gathers sufficient reputation, he gains competitive advantage in the market, so attracting customers is not a big issue. He thus may not be as prudent as before (not necessarily malicious) in some transactions, and bad feedbacks for latest transactions will not affect his reputation value greatly or immediately. In this chapter, we present our solution for detecting such imprudent behavior of a ‘reliable’ (considered by traditional reputation systems) agent. Firstly, by studying the Allegro dataset, we infer that a non-negligible fraction of unsatisfactory transactions involve sellers with high reputation, proving that such imprudent behaviors indeed broadly exist. Secondly, we propose the concept of *imprudence*, and identify/verify the factors that influence a seller’s imprudence behavior. We then combine these factors using logistic regression to predict and study the probability of imprudence of a target seller, and validate our results using real Allegro and eBay datasets.

This chapter is organized as follows: After the introduction section 8.1, our methodology is detailed in Section 8.2 where we first introduce used notations in Section 8.2.1 and then identify useful features for imprudence estimation in Section 8.2.2 before de-

scribing the proposed imprudence prediction strategy in Section 8.3. Training data collection and the learning and prediction algorithms are described in Sections 8.3.1, 8.3.2 and 8.3.3 respectively. Evaluation results are described in Section 8.4 where we validate the identified features, investigate the influence of various parameters, as well as compare the performance of our approach with some existing approaches. We finally draw conclusions and summarize future research directions in Section 8.5.

8.1 Introduction

Most online auction sites (*e.g.*, eBay, Taobao, Allegro) only deploy a simple reputation system where a buyer rates a seller according to his experience in the transaction. Such discrete ratings (*e.g.*, positive/neutral/negative) are aggregated to produce a reputation value for each seller. A lot of more advanced approaches [133, 19, 134, 80, 135, 20, 141] have also been proposed to detect dishonest sellers by studying past behavior of the specific seller in question. The principle of these solutions is to efficiently aggregate a seller’s historical information such that an accurate reputation value is generated. This reputation value is deemed to precisely reflect the reliability or trustworthiness of the target seller. Such approaches fail to discern the divergence of behavior of an entity with high reputation.

8.1.1 Motivation

In order to investigate the interplay of sellers’ behaviors and their reputation values, we study Allegro dataset. From a subset of 953 sellers with 71811 past transactions, we identified 648 transactions with neutral ratings and 1206 with negative ratings. We consider a transaction was satisfactory for the buyer only if its rating is positive, otherwise, if the rating is explicitly negative, or even neutral, it is deemed unsatisfactory or unsuccessful.

For each unsuccessful transaction, we calculate reputation value of the corresponding seller (*i.e.*, fraction of positive ratings) when this transaction happened. Figure 8.1 shows

the relationship between occurrence of an unsuccessful transaction and the reputation value using a cumulative distribution function. We observe that around 56.7% of the neutral ratings and 47.5% of the negative ratings are attributed to transactions of sellers with reputation values larger than 0.9. Moreover, there are still 18.8% of the neutral ratings and 11.0% of the negative ratings even when sellers’ reputation values are larger than 0.98.

Figure 8.1: Cumulative distribution function (CDF) of the reputation value for unsuccessful transactions

This study demonstrates that the reputation value of the seller in itself is not an adequate indicator of his trustworthiness (and thence highlight the goal of this thesis). In other words, current reputation mechanisms and the mainstream refinements, even if they are able to derive accurate reputation value (using direct transactions, third party opinions, or both), cannot provide reliable prediction of the target seller’s behavior, since a considerable fraction of unsuccessful transactions happen even when the corresponding seller’s reputation value is very high. On the other hand, although the work presented in Chapter 7 tries to detect the target agent’s behavior deviation without reputation, it applies the same method for all potential transactions, i.e., it does not differentiate the stages of an agent’s life cycle while the agent may change his behavior (intentional or unintentional) at different stages. Moreover, the accuracy of this work is greatly influenced by the sequence of the target agent’s past transactions, which, however, does not always hold in reality.

8.1.2 Problem Statement

From the discussion so far, we see the need for some metric, beyond reputation, i.e., beyond an average aggregate of an entity’s behavior, to more precisely predict its reliability. To that end, we propose the concept of *imprudence*, to describe such an ‘unexpected’ phenomenon: *imprudence of a seller refers to the situation where he behaves inappropriately (possibly out of complacency) when this seller is publicly considered reputable*. In our view, imprudence can be greatly interpreted by the Drive-Reduction Theory as well as basic motivation theories from the perspective of behavior science and psychology [144, 145, 146]. For instance, when a seller has already gathered sufficient reputation, it becomes easier for him to attract buyers. Therefore, he may not be as prudent as before to serve each buyer well since failures of several transactions will only have marginal effect on his reputation value or business opportunities. In that sense, gaining reputation may be seen as a second-order conditioning and extrinsic motivation of the seller. At the very beginning, the seller will behave prudently so as to gain appealing reputation and strengthen his capability to survive in the fierce competition of virtual marketplaces. However, as the reputation reaches a certain degree, the strength of such extrinsic motivation is reduced. Besides, it will lead to a subsequent reduction in intrinsic motivation influencing the seller’s behavior, of which the extrinsic and intrinsic motivation work together and make the seller perform seriously and consistently.

However, from the perspective of buyers, this inconsistent attitude and behavior of sellers might increase their uncertainty in decision making, costs and risks of transactions, even though a reputation mechanism is supposed to precisely reduce this kind of uncertainty and costs [147].

Therefore, the high level objective of this work is to detect inappropriate behavior of the ‘reliable’ (*i.e.*, reputable considered by traditional reputation systems) sellers to facilitate buyers’ decisions when encountering the sellers. We next outline our approach to study and identify the factors influencing individual sellers’ imprudence.

8.1.3 Approach

The general idea of our approach is to identify the possible features that are associated with imprudence of the sellers in the community. Such features are harnessed using data mining algorithms to extract the relationships between contexts of the imprudence of other sellers and that of encountered transaction/seller. In other words, we identify the imprudence pattern of other sellers in the community to estimate the probability of imprudence of the target seller.

By studying the Allegro dataset, we identify four features that are associated with the occurrence of imprudence of the ‘reliable’ sellers:

- Reputation value: from Figure 8.1 we observe that the densities of occurrence of unsuccessful transactions vary with reputation values (*i.e.*, steps in the figure);
- Number of past transactions before the current transaction: this feature influences the effects of reputation value, thus should also be considered. For instance, both (45 positive ratings, 5 negative ratings) and (9 positive ratings, 1 negative rating) produce reputation value of 0.9, but the former is more informative;
- Time difference between the first transaction and the current transaction: from behavior science, a seller may vary his behavior periodically or aperiodically;
- Item price: a seller may be imprudent in some transactions due to their low margins.

We then apply logistic regression to linearly combine these features to estimate the probability that the encountered seller will be imprudent in the near future. Logistic Regression is a kind of predictive model which can be used when the response variable is a categorical variable with two categories, *e.g.*, prudent/imprudent in our case. Compared to other linear regression models and classification algorithms (*e.g.*, linear discriminant analysis), logistic regression has several advantages like the response variable is bounded between 0 and 1; it is more robust; it relies on fewer assumptions, etc. Due to the fact

that positive ratings dominate the whole system, the estimated imprudence value is not reasonably high in absolute for direct imprudence prediction. We then propose a prediction algorithm by studying the trend of imprudence probabilities of a seller for his recent transactions to try to detect the potentially risky ones. Figure 8.2 depicts the workflow of our approach.

Figure 8.2: Imprudence detection workflow

Note that, different from traditional trust/reputation systems which try to distinguish dishonest sellers from honest ones, our approach is particularly designed to detect imprudent behavior of a ‘reliable’ seller (i.e., someone who will be traditionally classified as honest). Thus, our approach supplements and enhances the efficacy of traditional models.

8.2 Methodology

8.2.1 Notations

In an online auction site, we denote the set of buyers by $\mathcal{B} = \{B_1, B_2, \dots\}$ and the set of sellers by $\mathcal{S} = \{S_1, S_2, \dots\}$. For each seller S_i , his past transactions are denoted by $\Theta_i = \{\theta_{i,1}, \theta_{i,2}, \dots\}$. Table 8.1 summarizes the notations that will be used in this chapter.

8.2.2 Features

The core of this work is to identify features that are capable of distinguishing dishonest behavior when the target seller is already deemed ‘reliable’ by conventional reputation based mechanisms (i.e., reputation value is high and transaction history is relatively

Table 8.1: Summary of notations

Notations	Description
$\mathcal{B}$	Set of all buyers in the online auction site.
$\mathcal{S}$	Set of all sellers in the online auction site.
Θ_i	Set of past transactions of seller $S_i \in \mathcal{S}$.
$R_{i,j}$	Reputation value of S_i when his j^{th} transaction happened.
$N_{i,j}$	Number of S_i 's past transactions when his j^{th} transaction happened.
$T_{i,j}$	Time when S_i 's j^{th} transaction happened.
$P_{i,j}$	Price of the item in S_i 's j^{th} transaction.

long). Inspired by behavior science and psychology, as well as leveraging on observations from existing literature [148, 129], we study four potentially useful features, which will be discussed and verified in the next subsections. Note that there are possibly other useful features which can further improve the results, or replace some of the chosen ones. However, we emphasize that the selected features have already provided good prediction results (demonstrated in Section 8.4), and this work is primarily about the approach to look beyond reputation rather than identification of an exhaustive list of relevant features.

8.2.2.1 Reputation value

Most existing deployed reputation systems for online auction sites predict a seller's future behavior based on his reputation value, which is derived based on outcomes of his past transactions.

Figure 8.3.a shows the probability mass function (PMF) of sellers' reputation values for unsuccessful transactions, showing the variation of the frequency of unsuccessful transactions across different reputation values. Specifically, we show the fractions of unsuccessful transactions when the corresponding sellers' reputation values are within certain ranges (with 0.01 as the span). Since we are interested in ‘reliable’ sellers' behavior, so only the PMF for sellers with reasonably high reputation (≥ 0.75) is displayed. We observe that the fractions of unsuccessful transactions vary greatly with different reputation value ranges. Furthermore, we notice that the fractions are particularly high

Figure 8.3: Features verification

in certain high reputation ranges (*e.g.*, $[0.97, 0.98]$ and $[0.98, 0.99]$), which again demonstrates that reputation value may correlate with the probability of a seller’s imprudence.

8.2.2.2 Previous transactions volume

In some cases, a single reputation value may not be able to accurately reflect a seller’s past behavior. For instance, seller *A* has 20 successful transactions and 5 unsuccessful ones, and seller *B* has 80 successful transactions and 20 unsuccessful ones. Both sellers have the same reputation value $20/25 = 80/100 = 0.8$. However, compared to seller *B*, *A*’s reputation value is derived based on limited past transaction volume (*i.e.*, 25), thus suffering from low accuracy/confidence. So in addition to reputation value, it is also necessary to take into account the associated transaction volume.

In order to understand how the number of previous transactions (when a specific transaction happened) is correlated with a seller’s imprudence, we analyze the distribution of the previous unsuccessful transactions. Considering that different sellers may have diverse past transaction volumes, we do not directly study previous transaction

volume of a seller S_i . Instead, we study the relative position of a specific unsuccessful transaction $\theta_{i,j}$ in the seller S_i ’s life cycle, measured by $\rho_{i,j}$ as follows:

$$\rho_{i,j} = \frac{j}{|\Theta_i|} \quad (\text{Eq. 8.1})$$

where j indicates the transaction’s position in S_i ’s past transaction list. Note that we assume a seller S_i ’s past transactions follow chronological order throughout this chapter.

Figure 8.3.b shows the probability mass function (PMF) of ρ . We calculate that over 27.5% of the unsuccessful transactions happened within the sellers’ 20% most recent transactions. Moreover, we notice that at certain positions, the occurrence of unsuccessful transactions increases sharply (*e.g.*, when $\rho \in (0.85, 0.86)$, $(0.89, 0.90)$, $(0.95, 0.96)$, $(0.97, 0.98)$, etc.). Note that generally, a seller becomes more and more ‘reliable’. So, the fact that quite a few unsuccessful transactions nonetheless happened recently (and aperiodically) implies that this feature also correlates with a seller’s imprudent behavior and thus should be considered.

8.2.2.3 Time of the transaction happened

Similar to transactions volume, the time when a specific transaction happened is also expected to correlate with a seller’s imprudence, *i.e.*, a seller may change his behavior with the passage of time even though his reputation value is high. Due to the diversity of sellers’ system ages, it is not reasonable to directly study the relationship between time and a seller’s imprudence. Instead, we use the index of the relative time of a specific transaction in a seller’s life cycle. In particular, for a seller S_i , the index $\sigma_{i,j}$ is calculated as:

$$\sigma_{i,j} = \frac{T_{i,j} - T_{i,1}}{T_{i,|\Theta_i|} - T_{i,1}} \quad (\text{Eq. 8.2})$$

From Figure 8.3.c (PMF), we calculate that around 22.5% of the unsuccessful transactions happened even when σ is higher than 0.9 (*i.e.*, in very recent transactions). Moreover, in different time intervals (particularly, from 0.98 to 0.99), the distribution of unsuccessful transactions also varies. So the time of a transaction is also considered as a promising indicator of a seller’s imprudence.

Logically, this feature is similar to the previous feature. Nevertheless, we are still going to use both of them because, for each transaction, its position is not strictly linearly correlated with the time it happened. Thus, the two varied features can contribute to different contexts to influence sellers’ behaviors [149].

8.2.2.4 Item price

Sellers often adopt different marketing strategy and target at different potential buyers for products in different categories or price levels [150]. Hence, item price is an important factor that influences a seller’s behavior. For instance, a seller is more cautious in selling a laptop than in selling a pen. At the early stage, a seller is probably very cautious in every transaction, regardless the item price, to accumulate reputation. However, when his reputation is quite high, this seller may not concentrate on some transactions because even if such transactions are not satisfied by the buyers, his overall reputation and profit are not seriously affected.

In order to study the impact of item price on sellers’ imprudence, we first determine a threshold to distinguish unsuccessful transactions occurred when the corresponding sellers are ‘reliable’ (i.e., reputable by traditional reputation systems) from that occurred when the corresponding sellers are not yet ‘reliable’. We say a seller is ‘reliable’ when a transaction happened, if: (1) his reputation value is larger than 0.9 at that time, and (2) that transaction is one of his most recent transactions (*i.e.*, $\rho > 0.75$ and $\sigma > 0.75$). We use two metrics to compare ‘reliable’ sellers with ‘unreliable’ ones in terms of their unsuccessful transactions’ item prices: (1) average price under the same category and (2) difference between price of an unsuccessful transaction and the average price of the previous transactions under the same category. Figure 8.3.d shows the CDFs of the item prices and price differences by ‘reliable’ and ‘unreliable’ sellers respectively¹. It is obvious that for both metrics, they are higher for the ‘unreliable’ sellers than the ‘reliable’ ones. This again proves that a seller may become imprudent if he is selling a cheap item, or the item price is slightly different than the average market price. Note

---

¹To improve readability of the figure, only part of the price range is displayed.

that if an item price is largely different than the market price, failure of the transaction is very likely due to the seller’s maliciousness instead of imprudence. Recall that, this would be predicted by traditional models, while our approach focuses on detecting the imprudent behavior among ‘reliable’ sellers.

8.3 Prediction strategy

After having discriminating features, we need to combine them to predict the probability of a seller’s imprudence. To do so, we apply logistic regression, which is a kind of predictive models that can be used to model the probability of a categorical dependent variable, *e.g.*, prudent/imprudent. We next present our prediction strategy by starting with the collection of training data.

8.3.1 Training Data

Given a target seller, the general idea of our approach is to identify and learn the behavior pattern (*i.e.*, imprudence) from other sellers with similar profile. That is to say, we leverage the key observation that mass behavior can be used as an indicator to predict behavior of specific individuals [151]. It is however crucial to identify correctly the sellers that share similar properties with the target seller such that accurate training data can be obtained.

Following the features we studied in Section 8.2.2, given a specific seller S_i , his seller community is formed based on the following four criterion:

- *Reputation value*: Although reputation value is insufficient to determine the real trustworthiness of a seller, it indeed reflects a seller’s past behavior to a certain extend. We thus select the seller S_j satisfying $|R_i - R_j| < \epsilon_r$, where ϵ_r is the error threshold for reputation value;
- *Transaction volume*: As mentioned before, meaningfulness of a reputation value is influenced by the corresponding seller’s past transaction volume. So the seller

S_j with the similar number of past transactions is considered, i.e. $|N_i - N_j| < \epsilon_n$, where ϵ_n is the error threshold for transaction volume;

- *System age*: Similarly, the seller S_j with the similar system age should also be considered, i.e. $|T_i - T_j| < \epsilon_t$, where ϵ_t is the error threshold for system age;
- *Category*: Since a seller may behave differently in selling items in different categories, we select the sellers who are active in the same categories with the target seller. Note that since item price correlates with item category, we do not explicitly consider item price here.

Note that given a set of community formation criterion mentioned above, there is only one single community for a specific seller in the system. However, from the perspective of individual buyers, due to the factors such as resource constraints (particularly in a decentralized system), the communities they can access may be subsets of the objectively existent (one single) community. From this point of view, each buyer may have its personalized seller community for the target seller. However, doing so is costly, and we leave this for future work.

Once a community of sellers who are similar the target seller is formed, we obtain training data for logistic regression by collecting past transactions of each community member². Since we are interested in the behavior of ‘reliable’ sellers, we only collect a subset of the past transactions of the similar sellers. In particular, a past transaction of the sellers will be collected if it satisfies: (i) the corresponding sellers are considered reputable (i.e. with reputation value of at least 0.9) when the transaction happened; (ii) the transaction is one of the most recent transactions of the sellers (*e.g.*, $\rho \geq 0.75$); (iii) the transaction happened recently (*e.g.*, $\sigma \geq 0.75$). For details about how to calculate ρ and σ , refer to Section 8.2.2.3.

---

²Community size may influence prediction accuracy, *i.e.*, small amount of training data may not be able to reflect useful imprudence patterns. However, results reported in Section 8.4 demonstrate that current proposal can achieve reasonably high accuracy. We leave the deeper analysis of influence of community size for future work.

8.3.2 Learning Algorithm

After collecting selected community members’ past transactions and obtaining the values of the associated feature, we now apply logistic regression algorithm to predict the probability of the target seller’s imprudence.

Formally, we denote the variable of a seller S_i ’s imprudence in his transaction $\theta_{i,j}$ by P_i , given a set of features $F_{i,j} = \{f_{i,j}^1, f_{i,j}^2, \dots, f_{i,j}^k\}$ associated with $\theta_{i,j}$. In this work, $k = 4$. P_i equals to 1 if S_i acts imprudently in $\theta_{i,j}$ and 0 otherwise. The expected probability of imprudence then takes the form:

$$p_i = E(P_i = 1 | F_{i,j}) \quad (\text{Eq. 8.3})$$

Logistic regression tries to obtain the coefficients $b_m (0 \leq m \leq k)$ such that the *logit* of the probability is modeled as a linear function of the F_i , as follows:

$$\text{logit}(p_i) = \log\left(\frac{p_i}{1 - p_i}\right) = b_0 + \sum_{m=1}^k b_m f_{i,j}^m \quad (\text{Eq. 8.4})$$

So we obtain the probability p_i by:

$$p_i = \frac{1}{1 + e^{-(b_0 + \sum_{m=1}^k b_m f_{i,j}^m)}} \quad (\text{Eq. 8.5})$$

The coefficients $b_m (0 \leq m \leq k)$ are learned from the training data using maximum likelihood estimation, which tries to find the coefficients for which the probability of the observed data is maximized.

8.3.3 Prediction Algorithm

Using logistic regression, we can estimate the probability that a transaction with a ‘reliable’ seller will be unsuccessful. Since most transactions are successful, the estimated probability of imprudence is expected to be very low, so it is not advisable/practical to directly use such a probability value to make the prediction. Figures 8.4.a - 8.4.d show the estimated imprudence probabilities of the corresponding past transactions of some sellers with different representative unsuccessful past transaction volumes. Note that we

Figure 8.4: Evaluation of individual sellers

only show past transactions when a seller S_i is ‘reliable’ (*i.e.*, $R_i > 0.9$, $\rho_i > 0.75$ and $\sigma_i > 0.75$) and 1 on the x-axis stands for the first transaction when the corresponding seller became ‘reliable’. Accordingly, we also show the real outcomes correspond to the estimated imprudence probabilities (zero stands for the successful transactions and one stands for the unsuccessful ones on the right side y-axis).

It is clear from the figures that although the estimated probability is not reasonably high for direct imprudence prediction, we note and exploit the general trend that when an unsuccessful transaction happened, the corresponding estimated imprudence probability increased suddenly. We thus propose a threshold based heuristic algorithm for predicting a given transaction will be unsuccessful.

Specifically, when a buyer encounters a seller S_i , he first collects a set of S_i ’s past transactions, denoted by $\Theta'_i = \{\theta'_{i,1}, \theta'_{i,2}, \dots\}$, which happened when S_i is already ‘reliable’. We then define a threshold $\xi_i \in [\delta_{i,\downarrow}, \delta_{i,\uparrow}]$ for the buyer to estimate whether a sudden increase of the estimated probability stands for a potential unsuccessful transac-

tion, and $\delta_{i,\downarrow}$ and $\delta_{i,\uparrow}$ represent lower bound and upper bound of ξ_i respectively. Generally, for a potential transaction, if the estimated imprudence probability is exceptionally larger than that of the previous transaction, then the potential transaction is predicted as unsuccessful. To estimate $\delta_{i,\downarrow}$ and $\delta_{i,\uparrow}$, we denote the set of estimated imprudence probabilities corresponding to Θ'_i by $Prob_i = \{p_{i,1}, p_{i,2}, \dots\}$. Then $\delta_{i,\downarrow}$ is calculated as the average of difference between imprudence probability of a successful transaction and that of its previous transaction:

$$\delta_{i,\downarrow} = \frac{\sum |p_{i,k} - p_{i,k-1}|}{N_{i,\downarrow}} \quad (\text{Eq. 8.6})$$

where $p_{i,k}$ is the estimated imprudence probability of k^{th} transaction $\theta'_{i,k}$ whose outcome is successful and $N_{i,\downarrow}$ is the number of past successful transactions. Similarly, $\delta_{i,\uparrow}$ is calculated as:

$$\delta_{i,\uparrow} = \frac{\sum |p_{i,l} - p_{i,l-1}|}{N_{i,\uparrow}} \quad (\text{Eq. 8.7})$$

where $p_{i,l}$ is the estimated imprudence probability of the l^{th} transaction $\theta'_{i,l}$ whose outcome is unsuccessful and $N_{i,\uparrow}$ is the number of past unsuccessful transactions.

After estimating the lower bound $\delta_{i,\downarrow}$ and the upper bound $\delta_{i,\uparrow}$, we can set the threshold ξ_i between $\delta_{i,\downarrow}$ and $\delta_{i,\uparrow}$. An intuitive way to determine ξ_i is to use the average of $\delta_{i,\downarrow}$ and $\delta_{i,\uparrow}$:

$$\xi_i = \frac{(\delta_{i,\downarrow} + \delta_{i,\uparrow})}{2} \quad (\text{Eq. 8.8})$$

So for a specific transaction $\theta_{i,j}$ of seller S_i , if the difference between the estimated imprudence probability $p_{i,j}$ and that of the previous one $p_{i,j-1}$ is larger than or equal to the threshold ξ_i , seller S_i is predicted to be imprudent in this transaction.

8.4 Evaluation

In this section, we evaluate the effectiveness of our approach in detecting imprudence of the ‘reliable’ sellers using both Allegro and eBay datasets. We first discuss evaluation methodology in Section 8.4.1 and then present the results in Section 8.4.2.

8.4.1 Experiment Methodology

The first dataset used in the evaluation was collected from Allegro, a leading auction site in Eastern Europe. Please refer to Chapter 6.4.1 for detailed statistical data. We also crawled one of the most popular auction sites eBay for a more comprehensive study. We randomly selected 26,922 sellers from different categories and collected their all past transactions (969,213 with positive ratings, 1914 with neutral ratings and 3590 with negative ratings) conducted from April 10, 2000 to June 4, 2011.

Each transaction in both datasets is associated with a set of features that are discussed in Section 8.2.2. In order to fully understand behavior of a seller when he becomes ‘reliable’, we particularly select (661 for Allegro data and 12578 for eBay data) target sellers that have sufficient (at least 20) past transactions to study (the threshold of transaction amount will be studied in the experiments). For a potential transaction $\theta_{i,j}$, the corresponding seller S_i is ‘reliable’ if (1) his reputation value is larger than a certain reputation threshold; (2) the transaction is one of the most recent transactions (*i.e.*, $\rho_j > \text{a certain threshold}$) and (3) the transaction happened recently (*i.e.*, $\sigma_j > \text{a certain threshold}$). These thresholds that may influence a seller’s ‘reliability’ and thus affect the predicting precision and recall of our approach will be justified in the experiments.

For each target seller, we focus on predicting outcomes of the past transactions when he is already ‘reliable’. To do so, we first find the similar sellers to form a community³ (see Section 8.3.1 where we set $\epsilon_r = \epsilon_n = \epsilon_t = 0.025$). By studying past transactions of the community members, we estimate coefficients for logistic regression using the Matlab function *mnrfit*. Such coefficients are used to linearly combine features of a potential transaction to predict its outcome.

In order to fully validate efficacy of our approach, we compare it with the following existing approaches:

---

³Note that the target agent in one run may be one of the community members in another run. In other words, the testing data and training data may overlap if we evaluate multiple target sellers. However, in a specific run, a seller is either target seller or community member (training data and testing data is clearly separated). So the evaluation results of our approach are not affected by such ‘data mixture’.

- Traditional reputation system: A buyer judges the successfulness of a potential transaction simply based on the seller’s reputation value. The decision is made based on a threshold. If a seller’s reputation value is larger than the threshold, he is considered to be trustworthy in the potential transaction, and then the transaction will be successful. Otherwise, the transaction is predicted to be unsuccessful;
- Distrust model: A buyer calculates distrust value of the target seller as the fraction of his unsuccessful transactions. Similar to the above mentioned traditional reputation systems, the decision is also made based on the threshold. If a seller’s distrust value is larger than the threshold, he is considered to be untrustworthy, and the potential transaction will be unsuccessful. Otherwise, the transaction is predicted to be successful;
- StereoTrust based approach: This approach is based on our trust model StereoTrust [44]. Please refer to Chapter 6.3 for more details.
- Logistic regression (LR) based approach [152]: This approach classifies a potential transaction as successful or unsuccessful using logistic regression. Different from our approach, this approach models the probability of problematic transactions simply based on amounts of previous successful transactions and unsuccessful ones, without considering any features identified in our work.

The accuracy of each approach is measured by precision, *i.e.*, the actual fraction of predictions that correctly detect risky transactions, and recall, *i.e.*, the fraction of all unsuccessful transactions that are correctly detected. Moreover, given different scope of our approach, we also study algorithm coverage *i.e.*, fraction of the sellers/transactions on which our approach can work well.

8.4.2 Results

In the following subsections, we will demonstrate performance of our approach from different aspects. In Section 8.4.2.1, we verify different feature combinations to learn

which features indeed help distinguish imprudent behavior when a seller is deemed ‘reliable’. In Section 8.4.2.2, we study how our approach’s accuracy correlates with coverage under different situations (*i.e.*, different parameters). In Section 8.4.2.3, we compare our approach with several existing approaches.

Table 8.2: Feature verification. ‘R’ stands for reputation, ‘A’ stands for previous transactions volume, ‘T’ stands for time of the transaction happened and ‘P’ stands for item price.

Feature combination	Precision	Recall
R,A,T,P	0.4975	0.5779
A,T,P	0.0258	0.1108
R,T,P	0.4190	0.5452
R,A,P	0.4871	0.5976
R,A,T	0.9840	0.7337
R,A	0.9357	0.7126
R,T	0.7252	0.7099
R,P	0.5314	0.5444
A,P	0.0320	0.0621
A,T	0.0211	0.0919
T,P	0.0280	0.1608
R	0.3121	0.6953
A	0.0625	0.0992
T	0.0519	0.2052
P	0.0670	0.1326

8.4.2.1 Feature verification

We first demonstrate the performance of our approach with different inputs (feature combinations). Although we got similar results using both datasets, we only show results based on Allegro data. Table 8.2 summarizes the results under the condition that: the sellers with at least 50 past transactions are considered in the experiments; a seller S_i is ‘reliable’ if his reputation value is large than 0.6, $\rho_j > 0.6$ and $\sigma_j > 0.6$ when his j^{th} transaction happened. We observe that when all features are considered, our approach’s performance is not optimized. Instead, the combination of the features of reputation value, past transaction volume and time difference of the transaction happened yields the best result. This is likely because the extent the item price can

distinguish a ‘reliable’ seller’s behavior from an ‘unreliable’ one is not sufficient for accurate imprudence probability prediction. We also verified the feature combinations under different conditions (*i.e.*, various thresholds values) and obtained similar results. So in the following experiments, we use only the three-feature combination which yields the highest accuracy.

8.4.2.2 Accuracy versus coverage

Given the design goals of this work, we are only interested in ‘reliable’ sellers’ behavior. Such a scope makes our approach only applicable to a subset of the whole seller base (*i.e.*, seller coverage). Besides, by adjusting the parameters (*i.e.*, various thresholds) for ‘reliability’ determination, the amount of a seller’s transactions that are evaluated may be affected (*i.e.*, transaction coverage). On the other hand, accuracy of our approach may also be affected since input data has been changed. So, we next study how the accuracy of our approach correlates with its (seller/transaction) coverage, under different parameters.

We first look at the threshold of past transaction volume, which determines whether a seller has sufficient past experience or not (see Figure 8.5.a). It is obvious that the higher the threshold, the less the percent of sellers that can be predicted by our approach (*i.e.*, smaller seller coverage). On the other hand, with decreasing seller coverage, precision increases slowly, but recall, which does not change monotonously, reaches the peak at the threshold of 60.

Figures 8.5.b - 8.5.d show how different values of the three features affect the accuracy and transaction coverage of our approach⁴. For the threshold of reputation value, both precision and recall are not affected significantly, but slightly increase (for recall, after 0.7). For the thresholds of ρ and σ , precision, recall as well as transaction coverage decrease with the increasing thresholds. Particularly, the precision decreases sharply when σ is larger than 0.7. When the thresholds increase, the transaction coverage decreases sharply, and hence it is not easy to collect sufficient training data (see Section

---

⁴Note that the three features do not affect seller coverage.

Figure 8.5: Accuracy versus coverage (Allegro dataset)

8.3.1) for accurate estimation of imprudence probability compared to the reputation value threshold where the transaction coverage (obviously) does not change.

From the above results we can estimate empirically the optimal thresholds that give both good accuracy and coverage: previous transactions amount threshold is 50, reputation value threshold is 0.9, ρ threshold is 0.65 and σ threshold is 0.65 for the Allegro dataset. In the next section where we compare our approach with other approaches, these threshold values are used.

8.4.2.3 Comparison

We finally compare the performance of our approach with that of some existing approaches (refer to Section 8.4.1). Note that since these existing approaches finally derive the probability that the potential transaction is successful/unsuccessful, we thus need a threshold for decision making. We vary the threshold from 0 to 1 with 0.005 as the increment to obtain the optimal thresholds for highest precision and recall for each

Table 8.3: Performance comparison

Approaches		Precision (Allegro)	Recall (Allegro)	Precision (eBay)	Recall (eBay)
Our approach	(with person- alized training data)	0.9840	0.7337	0.8571	0.8673
	(with generic train- ing data)	0.5712	0.5855	0.5245	0.5910
Traditional reputa- tion system	(optimal global threshold)	0.7312	0.0747	0.7042	0.0817
	(optimal personal- ized thresholds)	0.8678	0.0801	0.8273	0.0925
Distrust model	(optimal global threshold)	0.7025	0.1040	0.6505	0.1231
	(optimal personal- ized thresholds)	0.7686	0.1258	0.7136	0.1513
StereoTrust based approach	(optimal global threshold)	0.7750	0.2905	0.7045	0.3318
	(optimal personal- ized thresholds)	0.7544	0.3102	0.6932	0.3649
LR based approach [152]	(optimal global threshold)	0.8602	0.0625	0.8055	0.1015
	(optimal personal- ized thresholds)	0.7950	0.0785	0.7245	0.1236

approach. We considered two threshold variants: optimal global threshold for all sellers and optimal personalized threshold for each specific seller. Recall that for each target seller, our approach clusters similar sellers to obtain the training data (*i.e.*, personalized training data). In order to fully demonstrate performance of our approach, we also show the results without collecting similar sellers. That is, for each seller, we randomly select a subset of sellers to obtain a generic training data.

From Table 8.3 we notice that our approach (personalized training data) provides the highest precision and recall using both Allegro data and eBay data. When only generic training data is used, the precision is lower than other approaches, but the recall is still much higher. The traditional reputation system and distrust model provide a reasonably high precision but they suffer from low recall (while distrust model is more accurate in terms of recall). Similarly, LR based approach has a quite high precision but also has

a low recall. StereoTrust is less accurate than our approach (with personalized training data) but outperforms the other three approaches in terms of recall.

The results can be interpreted as that different from our approach, other approaches only estimate probability of an unsuccessful transaction. However, given an overwhelming majority of successful transactions, no matter what methods are used, it is not easy to distinguish unsuccessful transactions from the successful ones. Instead, after identifying discriminating features that are correlated to imprudent behavior, our approach does not directly use the derived probability but studies the trends of the probabilities of the recent transactions to try to detect the problematic ones.

8.5 Conclusion

Existing works on trust and reputation predominantly infer reliability based on high reputation. While such approaches readily eliminates risky transactions with someone with poor reputation, they do not provide a finer grained decision support when interacting with someone with high reputation but nonetheless with a streak of intermittent unreliable actions.

From real online auction datasets, we first determined that imprudent behavior by someone with high reputation is indeed a problem that exists in practice, and accordingly developed an approach, which looks beyond the average behavior (as is typically done), instead focusing on a seller’s behavior at the specific stages of his life cycle, aided by carefully identified features. Different from existing approaches that directly use such a probability to make the predictions, we produce imprudence probabilities of the recent previous transactions and study the trend of these probabilities to detect the risky transactions.

A more comprehensive exploration of potential other features to improve predication accuracy, analysis of impact of community size, as well as use of other classification algorithms (than logistic regression) are part of our immediate future plans.

Publication(s)

- *Detecting imprudence of “reliable” sellers in online auction sites* (submitted to a conference). **Liu Xin**, Anwitaman Datta, Fang hui, Zhang jie

Part V

Concluding remarks and future directions

Chapter 9

Conclusion

9.1 Summary

Trust is an important ingredient facilitating reliable interactions among autonomous participants in diverse large-scale distributed systems e.g., Peer-to-Peer systems, e-commerce, grid, etc. In this thesis, we focus on developing novel trust mechanisms in such massively distributed environments to address drawbacks suffered by current reputation based approaches: (i) current solutions relies on the knowledge which may not be available to the assessor; (ii) current solutions fail to effectively model the target agent's dynamic behavior.

In order to address the first drawback, we propose to explore and utilize stereotypes to assess trustworthiness of the target agent whose past behavior information is not locally available to a trustor, which is very common in large scale, open distributed systems. A stereotype contains certain features of the agents and an expected outcome of the transaction. These features can be taken from the agents' profile information, or the agents' observed behavior. To build stereotypes, a trustor has to group other agents based on the features (e.g., agents who are interested in the same topic are grouped). These groups need not be mutually exclusive. Then, when facing an unknown agent, the trustor estimates the agent's trust using his stereotypes on the groups to which the unknown agent belongs. Such a stereotyping based approach serves as an alternative and complementary mechanism (than existing techniques) to compute trust even in

absence of (global) information that is likely to be unavailable/inaccurate in large-scale open systems.

In order to realize such a stereotyping based trust mechanism, in the second part of this thesis, Chapter 3 introduces StereoTrust, which used a beta distribution based approach to derive stereotypes. StereoTrust calculates stereotypes by taking binary ratings as input. This approach is based on computing trust scores by statistically updating beta probability density function (PDF). The posteriori stereotype score is computed by combining a priori stereotype score with the new transaction outcome. When assessing trust, the derived stereotypes are combined using weights measured by fractions of the transactions regarding the corresponding groups. We refine this basic StereoTrust model by taking into account third party opinions if any and propose an dichotomy based enhancement called d-StereoTrust. The essential idea is to create a “good” and a “bad” subgroup inside each stereotype. For each group/stereotype, the trustor assigns each one of his previous transaction partners to one of these subgroups based on his personal experience with the partner (e.g., the ratio of unsuccessful transactions). Then, the trustor uses the aggregated third parties’ opinions about the unknown target agent to determine how similar is the agent to the “good” and “bad” subgroup.

On the basis of StereoTrust, Chapter 4 proposes an improved trust framework MetaTrust that employs machine learning techniques to address an important issue influencing accuracy of the stereotype based trust assessment, which is ignored by StereoTrust model: facing all derived stereotypes, how to distinguish which stereotypes are more discriminating or useful than others. To that end, a trustor first divides his past transactions with other agents into two groups: the successful transaction group and unsuccessful transaction group. Then, based on the feature vector associated with each transaction, the trustor applies appropriate machine learning algorithms (e.g., linear discriminant analysis and decision tree in this thesis) to extract relationships between the potential transaction and the previous transactions. In addition, we also try to relax the restriction on the usage of MetaTrust (i.e., sufficient local knowledge is assumed) by

constructing a local knowledge sharing overlay network (LKSON) where inexperienced agents can gather necessary knowledge from other agents. Some practical issues such as how to construct LKSON, how to filter out useless information, how to protect privacy, what kind of local knowledge is shared and how to minimize network resources consumption, etc. are discussed.

In the third part of this thesis, in order to validate usefulness of the stereotyping based trust mechanisms, we apply StereoTrust and MetaTrust to two specific distributed settings, i.e., P2P storage systems and Internet auction sites. In Chapter 5, we propose a contextual trust based data placement scheme for P2P backup systems to achieve high data availability. Different from many existing data placement schemes that rely on peers' historical information, our scheme is particularly designed for the scenario where a specific peer's online pattern is not available by using a different kind of information (i.e., stereotypes) to estimate contextual trust (i.e., suitability) of unknown potential data holders. We consider two specific kinds of contexts, i.e., peers' time zone since many works have demonstrated strong geographic correlation in diurnal patterns of peers' uptime [109, 110]. Data holders in the same time zone will demonstrate similar online patterns thus providing more effective availability for the data owner. Likewise, peers within the same ISP typically experience higher throughput at an individual level, and this also confines most of the network traffic within a single autonomous system, providing prudent usage of system wide network resources. Our scheme is generic and complementary to existing approaches and thus can be integrated into existing P2P backup systems that use global placement schemes. The proposed scheme can also be viewed as an alternative to multi-objective optimization of the system.

In Chapter 6, we consider the problem of predicting trustworthiness of a potential transaction where the seller is unknown to the buyer in online auction sites. In our approach, a buyer builds stereotypes that aggregate and summarize the past experience according to certain attributes: (1) category of the item; (2) price of the item and (3) number of items already sold by the seller when the transaction occurs. Facing a potential transaction with an unknown agent, the buyer builds its trust by cumulating the

experience from the stereotypes to which the potential transaction conforms. Two kinds of knowledge can be used to form the stereotypes: individual buyer's local knowledge or community knowledge, which is a collection of many buyers' local knowledge. Generally, local knowledge produces the most accurate and personalized stereotypes. However, in case the buyers are inexperienced (very common in online auction sites), community knowledge can be used to derive the stereotypes. Although community stereotypes may not be as accurate as local stereotypes, they provide a complementary knowledge to evaluate the transaction.

In the fourth part of this thesis, we rethink the issues of the current reputation based trust mechanisms: the mainstream approach and its enhancements spend much effort in extracting accurate knowledge from various information sources. With the filtered information, trust is assessed following the rule of hand that the more successful the past transactions, the more trustworthy the target agent is considered, subject to any context constraints. However, such an approach fails to effectively assess trust of the intelligent agent who may strategically change behavior to maximize his profits. In order to solve this problem, in Chapter 7, we propose a trust prediction approach to study an agent's dynamic behavior by studying his long term past behavior. The past behavior patterns, which are identified based on featured described contexts matching, are used to guide trust assessment for the potential transaction with the target agent. In Chapter 8, we look at the specific stages of the target agent's life cycle to detect his imprudence behavior when he has already been considered 'reliable' by the existing reputation systems. Various features that are associated with an agent's imprudence are identified and harnessed using data mining algorithms to extract the relationships between contexts of the imprudence of other relevant agents and that of the encountered transaction/agent.

9.2 Future works

The approaches proposed by this thesis partly solve some challenging problems in trust research such as bootstrapping issue, dynamic behavior modeling, etc. However, there

are still several fundamental open issues and/or drawbacks that remain to be addressed, which are summarized as follows:

- All the approaches presented here rely on features for constructing stereotypes, detecting behavior patterns of individual agents, etc. However, we have not provided a mechanism to defend malicious agents who try to counterfeit some of these features, and hence render our approaches inaccurate.
- Our approaches utilize the agents' past transactions to assess trust. One underlying assumption is, no matter what kind of information is used, there is adequate such information for our approaches to work. In practice, one may need to estimate the minimal amount of the required information (e.g., number of past transactions) such that the trust assessment accuracy is guaranteed while the associated overheads are minimized.
- Although real datasets based evaluations show that our approaches offer reasonably high trust prediction accuracy, we have not conducted in depth sophisticated analysis on performance of these approaches. For example, a basic question is - "which approach, or combination of approaches is best to use?", the answer to which may have to be determined dynamically, depending on the specific action/transaction in question.

Besides the limitations mentioned above, there are a number of ways in which our trust mechanisms and their applications could be extended, to better aid decision making in open, large-scale distributed systems. In particular, we identify the following four areas for further exploration.

9.2.1 Extensions of the stereotyping based trust mechanisms

For StereoTrust, trust prediction accuracy needs to be further improved:

- Current StereoTrust relies on beta distribution, i.e., considers outcome of an interaction is binary. To cater to the more generic scenario, e.g., there are multiple levels of the interaction outcome, one possibility will be to use dirichlet distribution instead of beta distribution to model stereotypes.
- Each stereotype is associated with a feature vector. In some cases, trustor may want to develop new stereotypes that are associated with combined features. To handle this issue, it would be interesting to apply some machine learning algorithms (e.g., linear discriminant analysis) to produce new features by combining original features.
- In dynamic massively distributed systems, an agent's stereotypes may evolve over time. To better understand how stereotypes reflect trustworthiness of a specific agent, we intend to study how to model evolving stereotypes (e.g., using longitudinal study [153]).

For MetaTrust, besides linear discriminant analysis and decision tree, we wish to integrate more suitable machine learning algorithms to ensure that the trust framework caters to diverse settings.

Both StereoTrust and MetaTrust mainly rely on a trustor's local knowledge. This is quite helpful when a trustee's historical information is not available to the trustor. However, in some scenarios, the trustee's past behavior (direct experience and/or indirect experience) does exist in the system. It is thus interesting to investigate how to combine diverse trust information sources to further improve trust prediction accuracy. Mui et al. [31] provided Chernoff bound based approach to combine direct experience and indirect experience. Following this approach, we intend to develop novel solution to effectively combine stereotypes and the trustee's historical information.

9.2.2 More application areas

Besides P2P backup storage and online auction sites, we will explore more application scenarios for both StereoTrust and MetaTrust. For instance, it may be possible to

apply our trust mechanisms to filter out junk email or prioritize coming emails by studying features of the old emails. Or it is possible to apply our trust mechanisms to Web 2.0 sites such as Wikipedia or YouTube to recommend interesting contents to the users based on their preference (i.e., personalized recommendation), and study how the stereotypes based abstraction compares with traditional recommendation mechanisms such as collaborative filtering [154] and k-nearest neighborhood approach [155].

Another interesting area where trust can be used is the recently emerged cloud computing environment. In [156], we have reported on our first attempt to investigate how trust can promote better collaborations among cloud service providers to compose and provide complex services. In the future, we intend to apply our trust mechanisms to cloud computing extensively. For instance, in cloud storage, which comprises of multiple commodity machines, it could be possible to use trust mechanisms to help reasonably schedule network traffic and distribute storage load to improve resource utilization and reduce effects of machine failures.

9.2.3 Improving the proposed trust mechanisms for capturing an agent's dynamic behavior

In the fourth part of the thesis, we have demonstrated our first attempts to capture an agent's dynamic behavior. However, they have their limitations (e.g., relying on strong assumptions, etc.) thus should be improved accordingly for more realistic deployment. For the work described in Chapter 7, the accuracy of the current approach is greatly influenced by the order of the target agent's past transactions, which, however, does not always hold in reality. In the future work, we intend to relax such restriction. The possible solutions include introducing fuzziness in transaction windows matching algorithm or applying hidden markov model (HMM). For the work described in Chapter 8, a more comprehensive exploration of potential other features to improve the quality of prediction for the considered dataset, application of the general approach to other datasets and application domains, as well as use of other classification algorithms (than logistic regression) are part of our future plans. Moreover, one on-going work is to study

how user community (e.g., size, formation criterion, etc.) of the target agent influences the prediction accuracy.

9.2.4 Implementation and deployment

The other research direction is to implement our trust mechanisms to make them available in real world applications. One possible way to do so is to integrate StereoTrust, MetaTrust and our trust mechanisms aiming at capturing an agent's dynamic behavior into some existing trust library such as ProtoTrust [138], which is developed by our Polish colleagues for online auction sites. They have implemented a Firefox extension to test the library of algorithms. Such live deployment and testing can bring much value and deeper understanding of both user behaviors, as well as the trust models, and how decision support aided by such trust mechanisms lead to alteration of user behavior in turn, introducing a 'cat-and-mouse' dynamism in the system, and opening up new research challenges but also opportunities rooted in ground reality.

References

- [1] Facebook. <http://www.facebook.com/>.
- [2] eBay. <http://www.ebay.com/>.
- [3] Flickr. <http://www.flickr.com/>.
- [4] Twitter. <http://twitter.com/>.
- [5] Wikipedia. <http://www.wikipedia.org/>.
- [6] S. Androutsellis-Theotokis and D. Spinellis. A survey of peer-to-peer content distribution technologies. *ACM Computing Survey*, 36(4):335–371, 2004.
- [7] Kazaa. <http://www.kazaa.com>.
- [8] BitTorrent. <http://www.bittorrent.com/>.
- [9] eMule. <http://www.emule-project.net/home/perl/general.cgi?l=1>.
- [10] Seti@home. <http://setiathome.berkeley.edu/>.
- [11] Genome@home. <http://genomeathome.stanford.edu/>.
- [12] C. de Laat, G. Gross, L. Gommans, J. Vollbrecht, and D. Spence. Generic aaa architecture, rfc 2903. *Network Working Group*, 2000.
- [13] J. Weise. Public key infrastructure overview. *Sun Blueprints Online*, 2001.
- [14] L. Rasmusson, A. Rasmusson, and S. Janson. Reactive security and social control. In *Proceedings of the 19th National Information Systems Security Conference*, Baltimore, MD, USA, 1996.

- [15] Larue T. Hosmer. Trust: The connecting link between organizational theory and philosophical ethics. *The Academy of Management Review*, 20(2):379–403, 1995.
- [16] R. La Porta, F. Lopez de Silanes, A. Shleifer, and Robert W. Vishny. Trust in large organizations. NBER Working Papers 5864, National Bureau of Economic Research, Inc, December 1996.
- [17] A. Abdul-Rahman and S. Hailes. A distributed trust model. In *Proceedings of the 1997 workshop on New security paradigms (NSPW)*, pages 48–60, Langdale, Cumbria, UK, 1997.
- [18] D. Fahrenholtz and W. Lamersdorf. Transactional security for a distributed reputation management system. In *Proceedings of the Third International Conference on E-Commerce and Web Technologies (EC-WEB)*, pages 214–223, Aix-en-Provence, France, 2002.
- [19] A. Jøsang and R. Ismail. The beta reputation system. In *Proceedings of the 15th Bled Electronic Commerce Conference*, pages 324–337, Bled, Slovenia, 2002.
- [20] J. Zhang and R. Cohen. Evaluating the trustworthiness of advice about seller agents in e-marketplaces: A personalized approach. *Electronic Commerce Research and Applications*, 7:330–340, 2008.
- [21] K. Aberer and Z. Despotovic. Managing trust in a peer-2-peer information system. In *Proceedings of the tenth international conference on Information and knowledge management (CIKM)*, pages 310–317, Atlanta, Georgia, USA, 2001.
- [22] H. Liu, Ee-Peng Lim, Hady W. Lauw, Minh-Tam Le, Aixin Sun, Jaideep Srivastava, and Young Ae Kim. Predicting trusts among users of online communities: an epinions case study. In *Proceedings of the 9th ACM conference on Electronic commerce*, pages 310–319, Chicago, IL, USA, 2008.

- [23] S. Buchegger and Jean-Yves Le Boudec. The Effect of Rumor Spreading in Reputation Systems for Mobile Ad-hoc Networks. In *Proceedings of the 1st International Symposium on Modeling and Optimization in Mobile AdHoc and Wireless Networks (WiOpt)*, pages 131–140, Sophia-Antipolis, France, 2003.
- [24] S. Lee, R. Sherwood, and B. Bobby. Cooperative peer groups in nice. In *Proceedings of the 22nd INFOCOM*, pages 1272–1282, San Francisco, CA, USA, 2003.
- [25] Sepandar D. Kamvar, Mario T. Schlosser, and Hector Garcia-Molina. The eigen-trust algorithm for reputation management in p2p networks. In *Proceedings of the 12th international conference on World Wide Web (WWW)*, pages 640–651, Budapest, Hungary, 2003.
- [26] L. Xiong and L. Liu. Peertrust: supporting reputation-based trust for peer-to-peer electronic communities. *IEEE Transactions on Knowledge and Data Engineering*, 16(7):843–857, July 2004.
- [27] J. Weng, C.Y. Miao, and Angela Goh. An entropy-based approach to protecting rating systems from unfair testimonies. *IEICE - Transactions on Information and Systems*, E89-D:2502–2511, 2006.
- [28] R. Zhou and K. Hwang. Powertrust: A robust and scalable reputation system for trusted peer-to-peer computing. *IEEE Transaction on Parallel Distributed Systems*, 18(4):460–473, 2007.
- [29] N. Li, John C. Mitchell, and William H. Winsborough. Design of a role-based trust-management framework. In *Proceedings of the IEEE Symposium on Security and Privacy*, pages 114–130, Oakland, California, USA, 2002.
- [30] N. Li and John C. Mitchell. Rt: a role-based trust-management framework. In *Proceedings of DARPA Information Survivability Conference and Exposition*, pages 201–212, 2003.

- [31] L. Mui and M. Mohtashemi. A computational model of trust and reputation. In *Proceedings of the 35th Hawaii International Conference on System Science*, pages 2431–2439, Hawaii, USA, 2002.
- [32] P. Resnick and R. Zeckhauser. Trust among strangers in Internet transactions: Empirical analysis of eBay’s reputation system. In *The Economics of the Internet and E-Commerce*, volume 11 of *Advances in Applied Microeconomics*, pages 127–157. Elsevier Science, 2002.
- [33] R. Aringhieri, E. Damiani, S. De, Capitani Di Vimercati, S. Paraboschi, and P. Samarati. Fuzzy techniques for trust and reputation management in anonymous peer-to-peer systems. *Journal of the American Society for Information Science and Technology*, 57:528–537, 2006.
- [34] S. Liu, C.Y. Miao, Yin-Leng Theng, and Alex C. Kot. A clustering approach to filtering unfair testimonies for reputation systems. In *Proceedings of the 9th International Conference on Autonomous Agents and Multiagent Systems*, pages 1577–1578, Toronto, Canada, 2010.
- [35] Hady W. Lauw, Ee-Peng Lim, and Ke Wang. Bias and controversy: beyond the statistical deviation. In *Proceedings of the 12th ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 625–630, Philadelphia, PA, USA, 2006.
- [36] A. Jøsang, E. Gray, and M. Kinatader. Analysing topologies of transitive trust. In *Proceedings of the Workshop of Formal Aspects of Security and Trust (FAST)*, pages 9–22, Pisa, Italy, 2003.
- [37] R. Guha, R. Kumar, P. Raghavan, and A. Tomkins. Propagation of trust and distrust. In *Proceedings of the 13th international conference on World Wide Web*, pages 403–412, New York, NY, USA, 2004.

- [38] N. Ma, Ee-Peng Lim, Viet-An Nguyen, Aixin Sun, and Haifeng Liu. Trust relationship prediction using online product review data. In *Proceeding of the 1st ACM international workshop on Complex networks meet information and knowledge management*, pages 47–54, Hong Kong, China, 2009.
- [39] A. Ravichandran and J. Yoon. Trust management with delegation in grouped peer-to-peer communities. In *Proceedings of the eleventh ACM symposium on Access control models and technologies (SACMAT)*, pages 71–80, Lake Tahoe, California, USA, 2006.
- [40] J. Sabater and C. Sierra. Regret: reputation in gregarious societies. In *Proceedings of the fifth international conference on Autonomous agents (AGENTS)*, pages 194–195, Montreal, Quebec, Canada, 2001.
- [41] R. Bartoszynski and M. Niewiadomska-Bugaj. *Probability and statistical inference*. Wiley-Interscience, 2008.
- [42] Tom Mitchell. *Machine Learning*. McGraw Hill, 1997.
- [43] L. Mui, M. Mohtashemi, C. Ang, P. Szolovits, and A. Halberstadt. Ratings in distributed systems: A bayesian approach. In *Workshop on Information Technologies and Systems*, New Orleans, LA, USA, 2001.
- [44] X. Liu, A. Datta, K. Rzađca, and E.P. Lim. Stereotrust: a group based personalized trust model. In *Proceedings of the 18th ACM conference on Information and knowledge management (CIKM)*, pages 7–16, Hong Kong, China, 2009.
- [45] X. Liu, G. Tredan, and A. Datta. Metatrust: Discriminant analysis of local information for global trust assessment. In *Proceedings of the 10th International Conference on Autonomous Agents and Multiagent Systems (AAMAS)*, Taipei, Taiwan.

- [46] X. Liu, G. Tredan, and A. Datta. A generic trust framework for large-scale open systems using machine learning. *ArXiv e-prints*, March 2011.
- [47] Geoffrey J. McLachlan. *Discriminant Analysis and Statistical Pattern Recognition*. Wiley-Interscience, 2004.
- [48] J. R. Quinlan. Induction of decision trees. *Machine Learning*, 1(1):81–106, 1986.
- [49] C. Batten, K. Barr, A. Saraf, and S. Trepetin. pStore: A secure peer-to-peer backup system. Technical Memo MIT-LCS-TM-632, Massachusetts Institute of Technology, Laboratory for Computer Science, October 2002.
- [50] L.P. Cox, Christopher D. Murray, and Brian D. Noble. Pastiche: Making backup cheap and easy. In *Proceedings of the 5th symposium on Operating systems design and implementation (OSDI)*, pages 285–298, Boston, Massachusetts, USA, 2002.
- [51] L.P. Cox and Brian D. Noble. Samsara: honor among thieves in peer-to-peer storage. In *Proceedings of the nineteenth ACM symposium on Operating systems principles (SOSP)*, pages 120–132, Bolton Landing, NY, USA, 2003.
- [52] M. Lillibridge, S. Elnikety, A. Birrell, M. Burrows, and M. Isard. A cooperative internet backup scheme. In *Proceedings of the annual conference on USENIX Annual Technical Conference (ATEC)*, pages 29–41, San Antonio, Texas, USA, 2003.
- [53] M. Landers, H. Zhang, and Kian-Lee Tan. Peerstore: Better performance by relaxing in peer-to-peer backup. In *Proceedings of the Fourth International Conference on Peer-to-Peer Computing (P2P)*, pages 72–79, Zurich, Switzerland, 2004.
- [54] R. Bhagwan, K. Tati, Y.C. Cheng, S. Savage, and Geoffrey M. Voelker. Total recall: System support for automated availability management. In *Proceedings of the 1st conference on Symposium on Networked Systems Design and Implementation (NSDI)*, pages 337–350, San Francisco, California, USA, 2004.

- [55] Robbert van Renesse. Efficient reliable internet storage. In *Workshop on Dependable Distributed Data Management*, Florianopolis, Brazil, October 2004.
- [56] B.G. Chun, F. Dabek, A. Haeberlen, E. Sit, H. Weatherspoon, M. Frans Kaashoek, J. Kubiatowicz, and R. Morris. Efficient replica maintenance for distributed storage systems. In *Proceedings of the 3rd conference on Networked Systems Design & Implementation (NSDI)*, pages 45–58, San Jose, CA, USA, 2006.
- [57] I. Stoica, R. Morris, D. Karger, M. Frans Kaashoek, and H. Balakrishnan. Chord: A scalable peer-to-peer lookup service for internet applications. In *Proceedings of the 2001 conference on Applications, technologies, architectures, and protocols for computer communications (SIGCOMM)*, pages 149–160, San Diego, California, USA, 2001.
- [58] Antony I. T. Rowstron and P. Druschel. Pastry: Scalable, decentralized object location, and routing for large-scale peer-to-peer systems. In *Proceedings of the IFIP/ACM International Conference on Distributed Systems Platforms Heidelberg (Middleware)*, pages 329–350, Heidelberg, Germany, 2001.
- [59] S. Ratnasamy, P. Francis, M. Handley, R. Karp, and S. Shenker. A scalable content-addressable network. In *Proceedings of the 2001 conference on Applications, technologies, architectures, and protocols for computer communications (SIGCOMM)*, pages 161–172, San Diego, California, USA, 2001.
- [60] K. Aberer. P-grid: A self-organizing access structure for p2p information systems. In *Proceedings of the 9th International Conference on Cooperative Information Systems (CoopIS)*, pages 179–194, Trento, Italy, 2001.
- [61] Q. Xin, T. Schwarz, and Ethan L. Miller. Availability in global peer-to-peer storage systems. In *Proceedings of the 6th Workshop on Distributed Data and Structures (WDAS)*, pages 63–77, Lausanne, Switzerland, 2004.

- [62] S. Le Blond, F. Le Fessant, and E. Le Merrer. Finding good partners in availability-aware p2p networks. In *Proceedings of the 11th International Symposium on Stabilization, Safety, and Security of Distributed Systems (SSS)*, pages 472–484, Lyon, France, 2009.
- [63] D. N. Tran, F. Chiang, and J. Li. Friendstore: Cooperative online backup using trusted nodes. In *Proceedings of the 1st Workshop on Social Network Systems (SocialNets)*, pages 37–42, Glasgow, Scotland, 2008.
- [64] Allegro. <http://www.allegro.pl>.
- [65] D.H. McKnight and N.L. Chervany. The meanings of trust. MISRC Working Paper Series 96-04, University of Minnesota, Management Information Systems Research Center, 1996.
- [66] D. Gambetta. Can we trust trust? In D. Gambetta, editor, *Trust: Making and Breaking Cooperative Relations*, pages 213–237. Basil Blackwell, 1988.
- [67] Stephen P. Marsh. *Formalising Trust as a Computational Concept*. PhD thesis, University of Stirling, 1994.
- [68] R. T. Golembiewski and M. McConkie. The Centrality of Interpersonal Trust in Group Processes. *Theories of Group Processes*, pages 131–185, 1975.
- [69] T. Grandison and M. Sloman. A Survey of Trust in Internet Applications. *IEEE Communications Surveys and Tutorials*, 3(4):2–16, 2000.
- [70] B. Christianson and William S. Harbison. Why isn’t trust transitive? In *Proceedings of the International Workshop on Security Protocols*, pages 171–176, Paris, France, 1997.
- [71] A. Jøsang, R. Ismail, and C. Boyd. A survey of trust and reputation systems for online service provision. *Decision Support Systems*, 43(2):618–644, 2007.

- [72] A. Abdul-Rahman and S. Hailes. Supporting trust in virtual communities. In *Proceedings of the 33rd Hawaii International Conference on System Sciences (HICSS)*, Hawaii, USA, 2000.
- [73] Young Ae Kim, Minh-Tam Le, H.W. Lauw, Ee-Peng Lim, Haifeng Liu, and J. Srivastava. Building a web of trust without explicit trust ratings. In *Proceedings of IEEE 24th International Conference on Data Engineering Workshop*, pages 531–536, Chicago, IL, USA, 2008.
- [74] L. Page, S. Brin, R. Motwani, and T. Winograd. The pagerank citation ranking: Bringing order to the web. Technical Report 1999-66, Stanford InfoLab, November 1999. Previous number = SIDL-WP-1999-0120.
- [75] Michael J. Freedman, K. Lakshminarayanan, S. Rhea, and I. Stoica. Non-transitive connectivity and dhds. In *Proceedings of the 2nd conference on Real, Large Distributed Systems*, pages 55–60, San Francisco, CA, USA, 2005.
- [76] A. Das and M.M. Islam. A novel feedback based fast adaptive trust model for p2p networks. In *Proceedings of IEEE 35th Conference on Local Computer Networks (LCN)*, pages 552–559, Denver, Colorado, USA, 2010.
- [77] R. Nithyanand and K. Raman. Fuzzy privacy preserving peer-to-peer reputation management. *Cryptology ePrint Archive, Report 2009/442*, 2009.
- [78] Marc Sánchez-Artigas, Pedro García-López, and Blas Herrera. Exploring the feasibility of reputation systems under churn. *Communications Letters, IEEE*, 13(7):558–560, July 2009.
- [79] S. Buchegger and Jean-Yves Le Boudec. A robust reputation system for p2p and mobile ad-hoc networks. In *Proceedings of the Second Workshop on the Economics of Peer-to-Peer Systems*, pages 1–11, Cambridge, MA, USA, 2004.

- [80] W. T. L. Teacy, J. Patel, N. R. Jennings, and M. Luck. Travos: Trust and reputation in the context of inaccurate information sources. *Autonomous Agents and Multi-Agent Systems*, 12(2):183–198, 2006.
- [81] J. Zhang, R. Cohen, and K. Larson. A trust-based incentive mechanism for e-marketplaces. In *Trust in Agent Societies*, volume 5396 of *Lecture Notes in Computer Science*, pages 135–161. Springer Berlin / Heidelberg, 2008.
- [82] Elaine Rich. User modeling via stereotypes. *Cognitive Science*, 3:329–354, 1979.
- [83] Elaine Rich. Users are individuals: individualizing user models. *International Journal of Man-Machine Studies*, 18:199–214, 1983.
- [84] P. Resnick, N. Iacovou, M. Sushak, P. Bergstrom, and J. Riedl. Grouplens: An open architecture for collaborative filtering of netnews. In *1994 ACM Conference on Computer Supported Collaborative Work Conference*, pages 175–186, Chapel Hill, NC, 10/1994 1994. Association of Computing Machinery, Association of Computing Machinery.
- [85] Meisles A. Gleyzer Y. Ben-Shimon D. Shani G., Rokach L. A stereotypes-based hybrid recommender system for media items. 2007. AAAI2007 Workshop On Recommender Systems.
- [86] C. Burnett, Timothy J. Norman, and K. Sycara. Bootstrapping trust evaluations through stereotypes. In *Proceedings of the 9th International Conference on Autonomous Agents and Multiagent Systems*, pages 241–248, Toronto, Canada, 2010.
- [87] J. R. Quinlan. Learning with continuous classes. In *Australian Joint Conference on Artificial Intelligence*, pages 343–348, Hobart, Tasmania, 1992.
- [88] A. Jøsang, R. Hayward, and S. Pope. Trust network analysis with subjective logic. In *Proceedings of the 29th Australasian Computer Science Conference*, pages 85–94, Hobart, Australia, 2006.

- [89] C. Burnett, Timothy J. Norman, and K. Sycara. Sources of stereotypical trust in multi-agent systems. In *Proceedings of the Fourteenth International Workshop on Trust in Agent Societies*, pages 25–39, Taipei, Taiwan, 2011.
- [90] S. Tadelis. Firm reputation with hidden information. *Economic Theory*, 21:635–651, 2003.
- [91] J. Tirole. A theory of collective reputations (with applications to the persistence of corruption and to firm quality). *Review of Economic Studies*, 63:1–22, 1996.
- [92] X. Liu and A. Datta. Contextual trust aided enhancement of data availability in peer-to-peer backup storage system. *Journal of Network and Systems Management (accepted)*, 2011.
- [93] K. Fukunaga. *Introduction to Statistical Pattern Recognition*. Academic Press Professional, Inc., 1990.
- [94] L. Hyafil and R. L. Rivest. Constructing Optimal Binary Decision Trees is NP-complete. *Information Processing Letters*, 5(1):15–17, 1976.
- [95] J. Ross Quinlan. *C4.5: programs for machine learning*. Morgan Kaufmann Publishers Inc., San Francisco, CA, USA, 1993.
- [96] David J. C. MacKay. *Information theory, inference, and learning algorithms*. Cambridge University Press, 2003.
- [97] G. Swamynathan, Ben Y. Zhao, and Kevin C. Almeroth. Decoupling service and feedback trust in a peer-to-peer reputation system. In *Proceedings of International Workshop on Applications and Economics of Peer-to-Peer Systems (AEPP)*, pages 82–90, Nanjing, China, 2005.
- [98] Epinions. <http://www.epinions.com/>.

- [99] Kevin Hoffman, David Zage, and Cristina Nita-Rotaru. A survey of attack and defense techniques for reputation systems. *ACM Comput. Surv.*, 42:1:1–1:31, December 2009.
- [100] Lluís Pamies-Juarez, Pedro García López, and Marc Sánchez-Artigas. Rewarding stability in peer-to-peer backup systems. In *Proceedings of the 16th IEEE International Conference on Networks*, pages 1–6, New Delhi, India, 2008.
- [101] David A. Patterson, G. Gibson, and Randy H. Katz. A case for redundant arrays of inexpensive disks (raid). In *Proceedings of the ACM international conference on Management of data (SIGMOD)*, pages 109–116, Chicago, Illinois, USA, 1988.
- [102] Lluís Pamies-Juarez, Pedro García López, and Marc Sánchez-Artigas. Heterogeneity-aware erasure codes for peer-to-peer storage systems. In *Proceedings of the International Conference on Parallel Processing*, pages 412–419, Vienna, Austria, 2009.
- [103] A. Datta and K. Aberer. Internet-scale storage systems under churn - A steady state analysis. In *Proceedings of the Sixth IEEE International Conference on Peer-to-Peer Computing (P2P)*, pages 133–144, Cambridge, UK, 2006.
- [104] E. Sit, A. Haeberlen, F. Dabek, B.G. Chun, H. Weatherspoon, R. Morris, M. Frans Kaashoek, and J. Kubiatowicz. Proactive replication for data durability. In *Proceedings of the Fifth International Workshop on Peer-to-Peer Systems (IPTPS)*, Santa Barbara, CA, USA, 2006.
- [105] A. Duminuco, E. Biersack, and T. En-Najjary. Proactive replication in distributed storage systems using machine availability estimation. In *Proceedings of the ACM CoNEXT conference*, pages 1–12, New York, New York, USA, 2007.
- [106] Lluís Pamies-Juarez and Pedro García López. Maintaining data reliability without availability in p2p storage systems. In *Proceedings of the ACM Symposium on Applied Computing*, pages 684–688, Sierre, Switzerland, 2010.

- [107] R. Sharma, M. dell Amico A. Datta, and P. Michiardi. An empirical study of availability in friend-to-friend storage systems. In *Proceedings of the 11th IEEE International Conference on Peer-to-Peer Computing*, Kyoto, Japan, 2011.
- [108] J. Kubiatowicz, D. Bindel, Y. Chen, S. Czerwinski, P. Eaton, D. Geels, R. Gum-madi, S. Rhea, H. Weatherspoon, W. Weimer, C. Wells, and B. Zhao. OceanStore: An architecture for global-scale persistent storage. In *Proceedings of the Ninth International Conference on Architectural Support for Programming Languages and Operating Systems (ASPLOS)*, pages 190–201, Cambridge, Massachusetts, USA, 2000.
- [109] R. Bhagwan, S. Savage, and G. Voelker. Understanding availability. In *Peer-to-Peer Systems II*, volume 2735 of *Lecture Notes in Computer Science*, pages 256–267. Springer Berlin/Heidelberg, 2003.
- [110] John A. Chandy. Storage allocation in unreliable peer-to-peer systems. In *Proceedings of the International Conference on Dependable Systems and Networks*, pages 227–236, Philadelphia, PA, USA, 2006.
- [111] K. Rzađca, A. Datta, and S. Buchegger. Replica placement in p2p storage: Complexity and game theoretic analyses. In *The 30th International Conference on Distributed Computing Systems (ICDCS)*, pages 599–609, Genoa, Italy, 2010.
- [112] Traces. <http://www.cs.uiuc.edu/homes/pbg/availability/>.
- [113] P. B. Godfrey, S. Shenker, and I. Stoica. Minimizing churn in distributed systems. *Computer Communication Review*, 36(4):147–158, 2006.
- [114] A. G. Dimakis, P. B. Godfrey, Y. Wu, M. J. Wainwright, and K. Ramchandran. Network coding for distributed storage systems. *IEEE Transactions on Information Theory*, 56(9):4539–4551, 2010.
- [115] IPAddressGuide.com. <http://www.ipaddressguide.com/>.

- [116] A. Medina, A. Lakhina, I. Matta, and J. Byers. Brite: An approach to universal topology generation. In *Proceedings of the Ninth International Symposium in Modeling, Analysis and Simulation of Computer and Telecommunication Systems (MASCOTS)*, pages 346–353, Cincinnati, OH, USA, 2001.
- [117] B.M. Waxman. Routing of multipoint connections. *Selected Areas in Communications, IEEE Journal on*, 6(9):1617–1622, 1988.
- [118] D. Stutzbach and R. Rejaie. Understanding churn in peer-to-peer networks. In *Proceedings of the 6th ACM SIGCOMM conference on Internet measurement*, pages 189–202, Rio de Janeiro, Brazil, 2006.
- [119] X. Liu and A. Datta. Redundancy maintenance and garbage collection strategies in peer-to-peer storage systems. In *Proceedings of the 11th International Symposium on Stabilization, Safety, and Security of Distributed Systems (SSS)*, pages 515–530, Lyon, France, 2009.
- [120] K. Tati and Geoffrey M. Voelker. On object maintenance in peer-to-peer systems. In *Proceedings of the 5th International Workshop on Peer-to-Peer Systems (IPTPS)*, Santa Barbara, CA, USA, 2006.
- [121] Wuala. <http://wua.la/>.
- [122] I. Clarke, O. Sandberg, B. Wiley, and Theodore W. Hong. Freenet: A distributed anonymous information storage and retrieval system. In *International workshop on Designing privacy enhancing technologies*, pages 46–66, Berkeley, California, USA, 2000.
- [123] H. Cyrus and M. Blough Douglas. The effect of replica placement on routing robustness in distributed hash tables. In *Proceedings of the Sixth IEEE International Conference on Peer-to-Peer Computing (P2P)*, pages 57–66, Cambridge, UK, 2006.

- [124] S. Bernard and L. F. Fabrice. Optimizing peer-to-peer backup using lifetime estimations. In *Proceedings of the 2009 EDBT/ICDT Workshops*, pages 26–33, Saint-Petersburg, Russia, 2009.
- [125] H. Shen and C. Xu. Locality-aware and churn-resilient load-balancing algorithms in structured peer-to-peer networks. *IEEE Transaction on Parallel Distributed Systems*, vol. 18:849–862, 2007.
- [126] F. Giroire, J. Monteiro, and S. Perennes. P2p storage systems: How much locality can they tolerate? Technical Report 7006, NIRIA, 2009.
- [127] M. Jelasity and O. Babaoglu. T-man: Gossip-based overlay topology management. In *Proceedings of the Fourth International Workshop on Engineering Self-Organizing Applications (ESOA)*, pages 1–15, Hakodate, Japan, 2006.
- [128] J. Li and F. Dabek. F2f: Reliable storage in open networks. In *Proceedings of the 5th International Workshop on Peer-to-Peer Systems (IPTPS)*, pages 2–7, Santa Barbara, CA, USA, 2006.
- [129] X. Liu, T. Kaszuba, R. Nielek, A. Datta, and A. Wierzbicki. Using stereotypes to identify risky transactions in internet auctions. In *Proceedings of the IEEE Second International Conference on Social Computing*, pages 513–520, Minneapolis, Minnesota, USA, 2010.
- [130] Taobao. http://www.taobao.com/index_global.php.
- [131] 2007 IC3 Annual Report. http://www.ic3.gov/media/annualreport/2007_IC3Report.pdf, 2007.
- [132] T. Kaszuba, A. Hupa, and A. Wierzbicki. Comment classification for internet auction platforms. In *Local Proceedings of the 13th East-European Conference, ADBIS 2009*, pages 374–384, Riga, Latvia, 2009.

- [133] C. Dellarocas. Immunizing online reputation reporting systems against unfair ratings and discriminatory behavior. In *Proceedings of the 2nd ACM conference on Electronic commerce*, pages 150–157, Minneapolis, Minnesota, United States, 2000.
- [134] J. Radu and F. Boi. Minimum payments that reward honest reputation feedback. In *Proceedings of the 7th ACM conference on Electronic commerce*, pages 190–199, Ann Arbor, Michigan, USA, 2006.
- [135] J. Radu and F. Boi. Collusion-resistant, incentive-compatible feedback payments. In *Proceedings of the 8th ACM conference on Electronic commerce*, pages 200–209, San Diego, California, USA, 2007.
- [136] Y. Wang, L. Li, and Ee-Peng Lim. Price trust evaluation in e-service oriented applications. In *Proceedings of the 10th IEEE Conference on E-Commerce Technology and the 5th IEEE Conference on Enterprise Computing, E-Commerce and E-Services*, pages 165–172, Crystal City, Washington,D.C., USA, 2008.
- [137] C.E.H. Chua and J. Wareham. Fighting internet auction fraud: An assessment and proposal. *Computer*, 37(10):31–37, 2004.
- [138] T. Kaszuba, P. Turek, A. Wierzbicki, and R. Nielek. Prototrust: An environment for improved trust management in internet auctions. In *the 13th ADBIS*, pages 385–398, Riga, Latvia, 2009.
- [139] T. Kaszuba, K. Rządca, A. Wierzbicki, and G. Wierzowiecki. A blueprint for universal trust management services. Technical report, PJWSTK, Warsaw, Poland, 2008.
- [140] S. Marti and H. Garcia-Molina. Taxonomy of trust: Categorizing p2p reputation systems. *Computer Networks*, 50:472–484, 2006.

- [141] G. Lax and Giuseppe M. L. Sarne. Celltrust: a reputation model for c2c commerce. *Electronic Commerce Research*, 8:193–216, 2008.
- [142] X. Liu and A. Datta. A trust prediction approach capturing agents’ dynamic behavior. In *Proceedings of the 22nd International Joint Conference on Artificial Intelligent (IJCAI)*, Barcelona, Spain, 2011.
- [143] A. Jøsang and J. Haller. Dirichlet reputation systems. In *Proceedings of the 2nd International Conference on Availability, Reliability and Security*, pages 112–119, Vienna, Austria, 2007.
- [144] E. E. Coons and S. F. Leibowitz. Neal e. miller and his research. *Biofeedback*, 38(3):101–107, 2010.
- [145] O. H. Mowrer and L. N. Solomon. Contiguity vs. drive-reduction in conditioned fear: temporal variations in conditioned and unconditioned stimulus. *The American Journal of Psychology*, 67(1):26–38, 1954.
- [146] M. R. Lepper, D. Greene, and R. E. Nisbett. Undermining children’s intrinsic interest with extrinsic rewards: a test of the overjustification hypothesis. *Journal of Personality and Social Psychology*, 28(1):129–137, 1973.
- [147] C. Serarols, M. J. Co, and D. Spohn. A transaction cost approach to analysing industrial districts. *International Entrepreneurship and Management Journal*, 4:235–252, 2008.
- [148] D. H. Chau and C. Faloutsos. Fraud detection in electronic auction. In *European Web Mining Forum at ECML/PKDD*, Porto, Portugal, 2005.
- [149] S. Figge. Situation-dependent services - a challenge for mobile network operators. *Journal of Business Research*, 57:1416–1422, 2004.

- [150] S. Gupta and D. Lehmann. *Managing customers as investments: the strategic value of customers in the long run*, pages 70–77. Pearson Education/Wharton School Publishing, 2005.
- [151] B. D. Ziebart, A. Maas, J. A. Bagnell, and A. K. Dey. Human behavior modeling with maximum entropy inverse optimal control. In *AAAI Spring Symposium on Human Behavior Modeling*, 2009.
- [152] P. Resnick and R. Zeckhauser. Trust among strangers in Internet transactions: Empirical analysis of eBay’s reputation system. In Michael R. Baye, editor, *The Economics of the Internet and E-Commerce*, volume 11 of *Advances in Applied Microeconomics*, pages 127–157. Elsevier Science, 2002.
- [153] Judith D. Singer and John B. Willett. *Applied Longitudinal Data Analysis: Modeling Change and Event Occurrence*. Oxford University Press, 2003.
- [154] X. Su and Taghi M. Khoshgoftaar. A survey of collaborative filtering techniques. *Advances in Artificial Intelligence*, 2009:1–20, January 2009.
- [155] Badrul M. Sarwar, G. Karypis, Joseph A. Konstan, and J. Riedl. Application of dimensionality reduction in recommender system – A case study. In *Proceedings of ACM WebKDD Workshop*, pages 285–295, Boston, MA, USA, 2000.
- [156] X. Liu and A. Datta. On trust guided collaboration among cloud service providers. In *TrustCol, in conjunction with CollaborateCom 2010*, pages 1–8, Chicago, Illinois, USA, 2010.