A Trust Model with Statistical Foundation

JIANQIANG SHI

Thesis submitted to the
Faculty of Graduate and Postdoctoral Studies
In partial fulfillment of the requirements
For the M.Sc. degree in Systems Science

University of Ottawa

Table of Contents

List of Figures

List of Tables

Abstract

The widespread use of the Internet signals the need for a better understanding of trust as a basis for secure on-line interaction. In the face of increasing uncertainty and risk, users and machines must be allowed to reason effectively about the trustworthiness of other entities. In this thesis, we propose a trust model that assists users and machines with decision-making in online interactions by using past behavior as a predictor of likely future behavior. We develop a general method to automatically compute trust based on self-experience and the recommendations of others. Our trust model solves the problem of recommendation combination and detection of unfair recommendations. Our approach involves data analysis methods (Bayesian estimation, Dirichlet distribution), and machine learning methods (Weighted Majority Algorithm). Furthermore, we apply our trust model to several utility models to increase the accuracy of decision-making in different contexts of Web Services. We describe simulation experiments to illustrate its effectiveness, robustness and the evolution of trust.

Key Words

Trust, Utility, Decision making, Bayesian estimation, Weighted Majority Algorithm

Acknowledgements

I would like to express my sincere gratitude to my supervisors, Dr. Carlisle Adams and Dr. Gregor v. Bochmann, under whose supervision I chose this topic and began the thesis. I thank them for their trust. Without their support and encouragement, this work would not have been possible. I also thank them for their guidance. I thank them for their humor and impressive knowledge of matters ranging from mathematics to network security, to politics and beyond, and for sharing these with me.

My appreciation goes to my other committee members as well. To Dr. Liam Peyton and Dr. Thomas Tran for correlations and insightful comments. Their valuable feedback helped me to improve the thesis in many ways. A very special thanks goes out to Dr. Stephen Paul Marsh, who advised me and helped me in various aspects of trust definition and classification.

I would like to thank my friends at University of Ottawa: Zhili Qiao, Tongyan Zhu, and Kaining Wang for the fun, serious support, company, and wonderful talks. I would like to acknowledge my comrades in the AIAS (Advanced Internet Applications and Systems) Lab: Cheng Peng, Wei Zhang, Yingxia Yuan and Quanyou Zhou.

I cannot end without thanking my family. My parents, Baorong Wen (Mom) and Qingsheng Shi (Dad), have been a constant source of support. They have supported me in more ways than they realize. They have cared for me as efficiently from far away as they would have done from up close, and most of all, they have trusted me all along. I would like to thank my brother Jianyi Shi, with whom I shared so much growing up.

I thank my wife Lin Zhu for her constant encouragement, patience and understanding throughout.

The final word of thanks goes to Jiaxin, my shining little daughter. She has cheered me up inadvertently so many times.

This thesis is dedicated to my parents and brother, to my wife and daughter.

Jianqiang Shi

April 8, 2005

All models are wrong, but some are useful.
George Box, 1979

1 Introduction

1.1 Problem definition

With the expansion of the Internet, users and services are often required to interact with unknown entities. This is so in application areas such as e-commerce, knowledge sharing, and even online gaming. Because the entities are autonomous and potentially subject to different administrative and legal domains, it is important for each user to identify trustworthy entities or correspondents with whom he/she should interact, and untrustworthy correspondents with whom he/she should avoid interaction [8].

Trust models have emerged as an important risk management mechanism in online communities. The goal of trust models is to assist users with decision-making in online interactions, by using past behavior as a predictor of likely future behavior [12]. Most electronic marketplaces on the Internet, such as eBay, Yahoo Auction, Amazon, and Epinions, support some form of trust management mechanism. eBay, for example, encourages both parties of each transaction to rate one another with a positive (+1), a neutral (0), or a negative (-1) rating. eBay makes the cumulative ratings of its members publicly known to every registered user [12]. Epinions provides a mechanism to weave the “Web of Trust”, a network of members whose reviews and ratings have been consistently found valuable. Each member can write a review on any topic and product. Reviews can be rated as “Not Helpful”, “Somewhat Helpful”, “Helpful”, and “Very Helpful”. The “Web of Trust” mimics the way people share word-of-mouth advice every day. Shareaza, a P2P file sharing system, allows members to write comments and ratings with respect to shared files. Thus, Shareaza allows members to avoid those that are fakes and download good quality and accurately represented files.

According to Dellarocas [12], "despite their widespread adoption and undeniable importance, very little work has been published so far on the reliability of various trust mechanisms and, even more importantly, on ways in which these mechanisms can be compromised". Lack of trust has often been cited as a major barrier to the adoption of online interaction. To address this problem, a novel distributed trust model was developed.

Our goal is to develop a general trust model that can be used for making rational decisions in order to make optimal choices. It should be usable in the context of Web Services [15], [40] and online transactions that meet real people's needs. We have opted for a trust model that is based on stochastic models of Web Services. We will explain how trust can be built up from experimental evidence and how statistical methods can be applied, together with utility functions, to make rational choices between different service providers or different strategies for problem solving. Our trust model is scalable in the number of users and services, and is usable, both for people and artificial autonomous agents [19].

1.2 Overview of the research

This thesis describes my research interest for pursuing the Master of Science (M.Sc.) in Systems Science at University of Ottawa. The objective of this investigation is to examine the challenges involved in developing a general trust model in the context of Web Services [15], [40] to secure online interactions that meet real people's needs. It should be scalable in the number of users and services, and will be usable, both for people and artificial autonomous agents. This thesis has three primary research contributions:

1. A general framework for evaluating trust of entities and services in online communities. We develop a general method to automatically compute trust, based on self-experience and recommendations. The trust in our model is not a simple scalar, nor a binary variable, but a multi-dimensional variable. This flexibility allows its application to a variety of different usage scenarios.
2. A learning method to combine recommendations according to their statistical attributes. We use a machine learning method (the Weighted Majority Algorithm) in our model to handle recommendations. Our trust model can dynamically learn from

the history of recommendations. Our trust model can automatically estimate the quality of recommendations and therefore can make robust, accurate decisions.

3. An analysis of the characteristics of trust update. We study the dynamic of trust update based on self-experience and recommendations. We also study the trust of entities with evolving performance.

This work may facilitate online communities in two aspects:

1. Trust guides users in choosing products and services;
2. Trust assists autonomous agents to cooperate.

The rest of this thesis is organized as follows. Chapter 1 states the problem definition, introduces the motivation and goal of the research. Chapter 2 summarizes the background and related work. We first give definitions of key terms, including trust, situational trust, general trust, basic trust, and different entities. We review previous work on trust models, recommendation handling, and decision making (including the Expected Utility Theory). In chapter 3 we introduce the mathematical tools including the Bayesian estimation, the Dirichlet distribution, the Weighted Majority Algorithm. These tools are the fundamentals to our trust model. Chapter 4 and chapter 5 introduce our approach. We also present several simple examples to show how our trust model can be used in different contexts. Chapter 6 presents simulations and their results. We describe the different behavior of our trust model in several scenarios. The explanations to non-intuitive behaviors are also given. Chapter 7 concludes the thesis and points to future work. We discuss the limitations of our trust model and the ignored issues.

2 Background and related work

2.1 Definitions of important terms

This section briefly defines important terms that are used repeatedly in the thesis. By understanding the way these terms are used, the reader will gain a clearer understanding of the nature of the research.

2.1.1 Trust

Although we use trust every day of our lives, trust has suffered from an imperfect understanding, a plethora of definitions, and an informal use in the literature [36]. It is common to say "I trust you," but what does that mean? [36]

According to Merriam-Webster Online Dictionary [25], trust is:

1 a: *assured reliance on the character, ability, strength, or truth of someone or something;*

2 a: *dependence on something future or contingent;*

The ITU-T (International Telecommunication Union – Telecommunication Standardization Sector) Recommendation X.509 specification gives us one definition:

Generally, an entity can be said to "trust" a second entity when it (the first entity) makes the assumption that the second entity will behave exactly as the first entity expects.

Therefore, trust deals with assumptions, expectations, and behavior. This clearly implies that trust cannot be measured quantitatively, that there is risk associated with trust, and that the establishment of trust cannot always be fully automated (for example, when the entities are human users) [10]. In the above definitions, trust finally depends upon some sort of beliefs, predictions, or expectations. However, it is not clear where these beliefs and expectations come from [11].

Marsh [36] proposes the concept of trust to make agents less vulnerable to others. Trust is basic in any kind of action in an uncertain world; in particular it is crucial in any form of collaboration with other autonomous agents. There is no universally accepted definition for trust [36], [26]. Marsh's thesis [36] presents an overview of trust as a social phenomenon and discusses it formally. He argues that trust is:

1. *A means for understanding and adapting to the complexity of the environment.*
2. *A means of providing added robustness to independent agents.*
3. *A useful judgment in the light of experience of the behavior of others.*
4. *Applicable to inanimate others.*

Marsh's thesis argues these points from the point of view of artificial agents. Trust in an artificial agent is a means of providing an additional tool for the consideration of other agents and the environment in which it exists. Moreover, a formalization of trust enables the embedding of the concept into an artificial agent [36].

Elofson [17] gives a definition suitable for our purpose. He agrees that observations are important for trust, and he defines trust as follows: "*trust is the outcome of observations leading to the belief that the actions of another may be relied upon, without explicit guarantee, to achieve a goal in a risky situation*." Elofson notes that trust can be developed over time as the outcome of a series of confirming observations (also called the dynamics of trust). From his experimental work, Elofson concludes that information regarding the reasoning process of an entity, more than the actual conclusions of that entity, affect the trust in the conclusions of that entity [11].

Trust is formed and updated over time through direct interactions (self experiences) or through information (recommendations) provided by other entities of society about experiences they have had [26]. Each event that can influence the degree of trust is interpreted by the entity either as a negative or a positive experience. If the event is interpreted as a negative experience, the entity will lose his trust to some degree and if it is interpreted to be positive, the entity will gain trust to some degree. The change of degree of trust depends on the trust model used by the entity. This implies that the trusting entity carries out a form of continual verification and validation of the subject of trust over time [11].

2.1.2 Situational trust

According to Marsh [36], individual entities are members of the set of all entities, which is society, or community. A society of entities is defined as a number of entities which are grouped together according to some metric [36]. Entities can be members of several societies. For example, Alice is a member of the society of University of Ottawa, but also of the basketball team. Furthermore, Marsh [36] defines a situation as a point in time relative to a specific entity. Therefore, entities find themselves in particular situations by definition [36]. Different entities in the "same" situation will not consider it from the same point of view. Marsh [36] "illustrates this point with the story of children playing, some of whom have mud on their foreheads, and some who don't. For each child in the situation, it is different because each can see every other child except themselves [36]. Thus, in a particular situation, entities may have only incomplete knowledge about that situation, yet some knowledge may be common (there are some children with mud on their foreheads - which ones is more difficult to answer, since we don't know if we're one of them)." Marsh notes that situations are entity-centered or subjective. Following Marsh [36], we represent situations with Greek letters, with subscripts for the entity concerned: δ_α is situation δ from α's point of view, δ_β is situation δ from β's. Since it is evident which entity is doing the considering, the subscript is dropped in the formulas that follow.

In addition, according to Marsh [36], different situations will require different considerations with regard to trust, and most will come out with different values for trust, even in the same person; while I may trust Alice to drive me to the airport, I most certainly would not trust her to fly the plane! This suggests that entities should consider others relative to the situation in which they find themselves (indeed, to the ways in which they envisage the situation to involve). Thus, we have a representation for the amount of trust an entity has in another in a given situation [36]. This use of "situation" is similar to Abdul-Rahman and Hailes's use of "context" [3] and Tan and Thoen's use of "categories" [41].

In this thesis, we represent an entity's (α) trust in another entity (β) within a certain situation (δ) by the following expression: $T_\alpha(\beta,\delta)$. We note that the situation δ is from entity α's point of view.

We consider the trusting entity to be adaptive in the sense that he/she learns from past experience; we would then expect that situational trust is dependent on what has happened to the entity in the past. Good experiences lead to a greater situational trust in the trusted entity, and bad experiences lead to a lower situational trust [36].

2.1.3 General trust

General trust represents the trust entity α has in β. It is not relative to any *specific* situation, it simply represents overall trust in another entity [36]. Therefore, the general trust is derived from past experience with entity β in all situations. We write $T_{\alpha}(\beta)$ for the entity α's general trust in entity β. In Marsh [36], "the value represents the probability that entity β will behave 'as if' entity α trusts β. In other words, entity α expects that β will behave according to α's best interests, and will not attempt to harm α".

2.1.4 Basic trust

One may ask the question: What is the initial trust an entity may have without relying on recommendations or his/her direct experiences? This initial trust reflects the trusting disposition of the entity.

In Marsh [36], the initial trust is called "basic trust". Marsh [36] argues that "basic trust is derived from past experience in all situations, indeed through the entity's entire life time of experiences." Basic trust is simply representative of the general trusting disposition of the entity; it is not the amount of trust an entity has in any other entity, situation, or environment [36]. The higher the basic trust is, the more trusting the entity is. Considering entities as trusting entities allows us to simulate a basic 'disposition' to trust someone or something that has only just been encountered [36]. In Abdul-Rahman and Hailes [3], basic trust is the general trusting attitude of the entity, a pervasive attitude toward oneself and the world. We write T_{α} for the basic trust of entity α.

From the definition, we expect that basic trust is dependent on past experience. Good experiences lead to a greater disposition to trust, and vice versa [36]. Marsh [36] also presents two pathological forms of basic trust: optimists and pessimists. The optimists' basic trust in others can only increase, despite past experience. The opposite is true for pessimists [36]. In this thesis, we assume the basic trust is fixed for simplicity.

2.1.5 Different types of entities involved in the trust model

Basically, we can identify three types of entities involved in a trust model: (1) the trusting entity, which we will represent by the letter α, (2) the trusted entity, which we will represent by the letter β, and (3) possibly some recommender entities. The trusted entity is usually a kind of service provider. Typically there are several service providers and the trusting entity, a potential client, has to decide which service provider to choose in order to satisfy some existing needs. The purpose of the trust model is to facilitate this decision. If the trust of the entity α is not well established (e.g. the entity α has few direct experiences upon which to draw), the entity may rely on recommendations from recommender entities. A recommender entity could be of the same nature as the client α, or it could be a recommendation server that accumulates information over time from many recommenders [19].

We also consider trust from the perspective of the software agent. In this thesis, the concept "user", "agent", and "entity" are exchangeable [19].

2.2 Review of trust models

In the last few years, there has been a lot of work on trust within the context of e-commerce and Internet security. We provide an overview of some significant work in the following.

The trust model proposed by Marsh [36] integrates many aspects of trust taken from sociology and psychology. Marsh [36] chooses to represent trust as a continuous variable over the range [-1, 1). The higher the trust value, the more trustworthy the entity is. His formalization of trust provides the basis for trusting artificial agents, which could form stable coalitions, take "knowledgeable" risks, and make robust decisions in complex environments. Marsh [36] presents experimental results from simple implementations of artificial trusting agents. Marsh's model has strong sociological foundations. Marsh tries to incorporate all aspects of social trust and introduces a large number of variables into his model. It is a relatively complex and abstract model and cannot be easily used in today's electronic communities. As Marsh [36] pointed out, several limitations exist for his trust model. Firstly, trust is represented in his model as subjective real number between the arbitrary range -1 and +1. The model exhibits problems at the extreme values and at 0. Secondly, the operators and algebra for manipulating trust values are limited and have trouble dealing with negative trust values. Marsh also pointed to difficulties with the concept of "negative" trust and its propagation [22].

Abdul-Rahman and Hailes [3] propose a trust model based on subjective probability.

They propose that the trust concept can be divided into *direct* and *recommender* trust. They represent direct trust as one of four agent-specified values about another agent ("very trustworthy", "trustworthy", "untrustworthy", and "very untrustworthy"). Recommended trust can be derived from word-of-mouth recommendations, which Abdul-Rahman and Hailes consider as "reputation". The translation from recommendations to trust is performed through an *ad-hoc* scheme [22]. Their aim is to provide a trust model based on the real world social properties of trust, founded on work from the social sciences. Trust is not an objective property of an agent, but a subjective degree of belief about agents. A trust decision is based on the truster's relevant prior experiences and knowledge. They introduce "semantic distance" to indicate the difference between a recommendation and a truster's own perception [3].

Yu and Singh [8], focusing on acquiring information from other agents in an agent community, propose a trust model which adapts the Dempster-Shafer Theory of Evidence [16] to represent and propagate the ratings that agents give to their correspondents. When evaluating the trustworthiness of a correspondent, an agent combines its local evidence (based on direct prior interactions with the correspondent) with the testimonies of other agents regarding the same correspondent. Their approach includes the TrustNet representation through which the ratings can be combined. In their following work [9], Yu and Singh introduce the weighted majority technique for belief function and aggregation to detect deceptive testimonies.

Tran and Cohen [37] propose reputation-oriented reinforcement learning algorithms for buyers and sellers in open, dynamic, uncertain, and untrusted market environments. In their approach, buyers learn to avoid the risk of purchasing low quality goods and to maximize their expected value of goods by dynamically maintaining sets of reputable and disreputable sellers. Sellers learn to maximize their expected profits by adjusting product prices and optionally altering the quality of their goods. A buyer selects a seller based on his/her direct experiences without communicating with other buyers.

Chen and Singh [24] propose a hierarchical reputation structure and the confidence level of reputation. The evaluation of an individual comment is split into two parts: evaluation of its numeric rating, and evaluation of its text review. A reputation tree is built, which consists of local match, global match, and confidence. They develop a general method to compute reputations for raters based on the ratings they and others give to objects, and incorporate these reputations to generate value-added information about rated objects. The value-added information includes the reputations of raters (organized as a hierarchy of categories), the scores of objects (taking reputations of raters into account) and the confidence level of scores.

2.3 Review of methods for dealing with Recommendations

We consider the problem of distributed trust management in large distributed systems of autonomous and heterogeneous entities. The basic idea is to let entities rate each other's performance during transactions. The aggregated ratings about a given entity are used to derive trust, which can assist an entity in deciding whether or not to transact with the given entity in the future. Furthermore, entities can exchange information regarding the aggregated ratings of a given entity via recommendations [4]. In such trust referral systems, it is then the summation of recommendations that plays an important role in decision making.

In trust referral systems, a fundamental problem is that the quality of recommendations is not guaranteed, in the sense that malicious entities could give unfair recommendations and/or the subjective evaluation criteria may be different. Because entities are distributed and autonomous, it is generally unreasonable to assume that there exist universally accepted trustworthy authorities who can declare the trustworthiness of different entities [9]. Consequently, the trusting entities must rely on themselves for discerning the trustworthiness of recommenders and the quality of recommendations.

Dellarocas [12] presents an approach which is targeted at detecting and excluding ratings that are unfairly high (he calls this “ballot-stuffing”). Dellarocas proposes mechanisms for addressing two important classes of reputation system fraud: (1) buyers intentionally provide unfairly high or unfairly low ratings for sellers; (2) sellers attempt to hide behind their cumulative reputation in order to discriminate on the quality of service they provide to different buyers (positive discrimination, negative discrimination). Dellarocas uses controlled anonymity to avoid unfairly low ratings and negative discrimination, and uses cluster filtering to reduce the effect of unfairly high ratings and positive discrimination [4]. To calculate an unbiased personalized reputation estimate, Dellarocas' scheme first uses collaborative filtering to determine a neighborhood group of buyers whose ratings over commonly rated sellers are similar. Then the ratings over a particular seller are divided into two groups (upper and lower cluster) by the Macnaughton-Smith *et al*. (1964) cluster filtering algorithm [29] in order to filter out the ratings that are most likely to be unfairly high [4]. The final reputation estimate is calculated from the ratings in the lower cluster only. Under the assumption that the fair and unfair ratings follow the same distribution type (e.g. normal distribution), but with different parameters, this technique typically reduces the unfair bias by more than 80% [4].

Yu and Singh [9] developed a model of reputation management based on the Dempster-Shafer Theory of Evidence [16]. Their approach focused on detecting and protecting against spurious ratings. Their approach involves an application of the Weighted Majority Algorithm to the Dempster-Shafer belief function and aggregation. Yu and Singh [9] distinguish between local belief and total belief. Local belief is from direct interactions and can be propagated to others. Total belief is the combination of local belief (if any) and testimonies received from any witnesses (recommenders). A TrustNet (TN) is built on an agent's acquaintances and neighbors and trustworthiness testimonies (recommendations) are collected by branching out through the referral chain (the references to other agents is controlled by a branching factor and depth limit). The testimonies of these witnesses are combined by using the Weighted Majority Algorithm to improve prediction accuracy. Some simple models of deception are studied to show the prediction accuracy and the evolution of trust networks [9].

Whitby *et al*. [4] propose a statistical filtering technique for excluding unfair ratings in Bayesian Reputation Systems. The assumption in their approach is that unfair ratings can be recognized by their statistical properties only. Whitby *et al*. design an iterated filtering algorithm based on the Beta distribution for binary ratings systems to exclude the presumed unfair ratings. In Whitby *et al.* [4], "the filtering algorithm is executed whenever an agent Z's reputation score must be recalculated. It assumes the existence of cumulative ratings vectors ρ(X,Z) for each rater X in the community. The basic principle is to verify that the score R(Z) of an agent Z falls between the *q* quantile (lower) and (1-*q*) quantile (upper) of ρ(X,Z) for each rater X. Whenever that is not the case for a given rater, that rater is considered unfair, and its ratings are excluded" [4]. The simulation results demonstrated that the filtering algorithm is most effective when a moderate (less than 30%) number of raters behave consistently unfairly [4].

We developed a model of trust management based on the Bayesian estimation and the Weighted Majority Algorithm (WMA). In the WMA, the weight given to each recommender depends on the history and quality of the recommendations given by the recommender. The assumption is that recommenders with low weights are likely to give unfair recommendations and recommenders with high weights are likely to give fair recommendations. The weights are updated after each transaction. By assigning different weights, our proposal detects and protects against unfair recommendations. By using Bayesian estimation, our proposal integrates the subjective prior knowledge and the actual experience into the estimation of trust.

Our proposal is different from the above proposals. Unlike Yu and Singh's proposal [9], our proposal uses the Bayesian estimation and integrates the subjective prior knowledge into the estimation of trust. Our proposal also extends univariate trust ratings to multivariate ratings [19]. Furthermore, we focus on recommendation combination, and deliberately ignore the problem of finding witnesses. With respect to Whitby *et al.*'s proposal [4], our proposal extends binary ratings systems to multinomial outcome systems using the Dirichlet distribution and takes self experience into account to calculate trust. Whitby *et al.*'s proposal is a memoryless system[1], which calculates the combined rating based on current recommendations only, while entities in our trust model maintain a set of weights correlated with past recommendations, and past experience.

[1] In a time-domain system with input x and output y, if the output y at each time depends only on the input x at that time, then such systems are said to be memoryless because you do not have to remember previous values of the input in order to determine the current value of the output [14].

2.4 Review of methods for decision making

Even though trust greatly reduces the complexity of society, we seldom make decisions solely based on the trust value. Rather, we believe that we typically make decisions based on utility we expect to gain. Decision making is often a question of selecting the optimal choice among a number of alternatives. It is therefore important to understand how different alternatives are evaluated in order to determine which is optimal. This means that for each alternative, a utility must be defined so that the alternative with the highest utility can be chosen. These kinds of approaches have been used in different areas. We provide an overview of some significant work in the following.

Expected Utility Theory (EUT) [31] states that the decision maker (DM) chooses between risky or uncertain prospects by comparing their expected utility values, i.e., the weighted sums obtained by adding the utility values of outcomes multiplied by their respective probabilities. The most popular expected utility function is the linear compensatory model in which preference for a product or service is represented by $x_j = \sum_{k=1}^{K} w_k y_{jk}$ where x_j is the preference for a product or service j, y_{jk} is the amount of attribute k in product or service j, and w_k is the importance weight assigned to attribute k [20]. The utility captures the DM's preference. The DM should choose the product or service that maximizes the expected utility.

In quality of service negotiation [5], a user satisfaction function plays a similar role to the utility function. The overall satisfaction of a product or service is expressed as a function of the individual component satisfaction s_i. That is $S_{total}=f(s_1,s_2,...,s_n)$. Desirable properties of $f(.)$ include:

1. If s_i is much smaller than each of others $\{s_1,s_2,...,s_n\}$, then S_{total} must be dominated by s_i.
2. $S_{total}=f(s,s,...,s)$ must be equal to s which allows $f(s_1,s_2,...,s_n)$ to be scaled.

One relationship that satisfies these requirements is

$$S_{total} = \frac{n}{\sum_{i=1}^{n} \frac{1}{s_i}} \text{ or } \frac{1}{\mathrm{S}_{\mathrm{total}}} = \frac{1}{n} \times \sum_{i=1}^{n} \frac{1}{s_i}$$

Note that this formula is similar to the formula for the equivalent resistance of parallel resistors, except that the result is multiplied by *n* so that requirement 2 is satisfied [5]. An entity will choose the product or service with the maximum satisfaction value S_{total}.

We believe that every trust model faces the decision making problem. Unfortunately, not every trust model explicitly points out the decision making process partly because of some of them are quite simple and straightforward. We provide an overview of some trust models in the following:

In Marsh [36], the decision making is simplified to the decision of cooperation. Having determined the trust in *β*, *α* can consider whether or not to cooperate. In other words, does *α* trust *β* enough to cooperate with her/him? To determine the answer to this question, Marsh introduces the concept of threshold values for trust. If the trust is above a cooperation threshold, cooperation will occur. If not, cooperation will not occur [36]. Marsh [36] points out that this view of cooperation may seem somewhat simplistic; however, it presents a straightforward notion of cooperation. In addition, it provides an interesting view of the problem which is not so complex as to put substantial overhead onto an implementation. In his later section, Marsh points out that the cooperation threshold is a subjective measure.

In the market simulation of Whitby *et al*. [4], the market consists of several sellers and buyers who trade in a single commodity. At the end of each round, the buyers and sellers adapt their behavior based on their previous rounds. Seller behavior is determined by two parameters, a selling price and an honesty level. Once the seller has committed to a transaction, it will either ship the item with a probability equal to the honesty level, or it will not ship the item. Thus the seller's transaction gain is different for an honest and a dishonest transaction. The seller will adapt its price and honesty level to maximize its gain. The following is the buyer behavior. For a transaction with a given seller *S*, the buyer *B* expects to receive the expected gain *G(B,S)*. The buyer will search for the seller that maximizes *G(B,S)* with a probability equal to the search propensity. The rest of the time the buyer will transact with the first seller whose price does not exceed the buyer's utility (All buyers receive the same utility from an item).

Jøsang [6] describes subjective logic as a logic which operates on subjective beliefs about the world, and uses the term opinion[2] to denote the representation of a subjective belief. He proposes that the opinions can be ordered according to probability expectation value, but additional criteria are needed in case of equal probability expectation values. Jøsang [6] defines the following order of opinion:

1. The opinion with the greatest probability expectation[3] is the greatest opinion.
2. The opinion with the least uncertainty is the greatest opinion.
3. The opinion with the least relative atomicity[4] is the greatest opinion.

The first criterion is self evident. The second is less so, but it is supported by experimental findings [6]. Jøsang admits that the third criterion is more an intuitive guess. Several applications of these criteria are illustrated by examples.

3 Mathematical tools

In this chapter, several mathematical tools will be briefly introduced. These tools are used in our trust model. Each tool is self contained and independent.

[2] In Jøsang [6], let Θ be a binary frame of discernment with two atomic states x and $\neg x$. Let *b(x), d(x), u(x),* and *a(x)* represent the belief, disbelief, uncertainty and relative atomicity functions on x in 2^{Θ} respectively. Then the opinion about x, denoted by w_x, is the tuple defined by:

$w_x \equiv (b(x),d(x),u(x),a(x))$.

[3] In Jøsang [6], the probability expectation is $E(x) = \sum_{y} m_{\Theta}(y) a(x/y),\ y \in 2^{\Theta}$

belief function is $b(x) = \sum_{y \subseteq x} m_{\Theta}(y),\ x, y \in 2^{\Theta}$

disbelief function is $d(x) = \sum_{y \cap x = 0} m_{\Theta}(y),\ x, y \in 2^{\Theta}$

uncertainty function is $u(x) = \sum_{\substack{y \cap x \neq 0 \\ y \not\subseteq x}} m_{\Theta}(y),\ x, y \in 2^{\Theta}$

[4] In Jøsang [6], relative atomicity: $a(x/y) = \dfrac{|x \cap y|}{|y|}$ where $x, y \subset 2^{\Theta}, \Theta$ is a frame of discernment.

A frame of discernment delimits a set of possible states of a given system, exactly one of which is assumed to be true at any one time. The relative atomicity is a value in the range [0,1].

3.1 Bayesian estimation

3.1.1 Introduction

Bayesian estimation is a popular representation for encoding uncertain expert knowledge in expert systems [13]. Bayesian statistics provide a conceptually simple process for updating uncertainty in the light of evidence. Initial beliefs about some unknown quantity are represented by a *prior* distribution. The *prior* distribution and information in the data are then combined to obtain the *posterior* distribution for the quantity of interest. The *posterior* distribution expresses the revised uncertainty in light of the data; in other words, an organized appraisal after consideration of previous experience [35].

Compared with classical approaches, Bayesian methods have at least the following advantages [13]:

Bayesian statistical techniques facilitate the combination of domain knowledge and data. Anyone who has preformed a real-world analysis knows the importance of *prior* or domain knowledge, especially when data is scarce or expensive. The *prior* knowledge and data can be combined with well-studied techniques from Bayesian statistics [13].

Bayesian methods offer an efficient and principled approach for avoiding the over fitting of the data - it models the training sequences very well, but will not fit other sequences from the same family. This is particularly likely if there are few training sequences. To avoid this problem a *regularizer* can be used, and in Bayesian methods it is tightly connected with the *prior* distribution [33]. There is no need to hold out some of the available data for testing. Using the Bayesian approach, models can be "smoothed" in such a way that all available data can be used for training [13].

The mathematical analysis leading to the expression for *posterior* distribution can be found in many text books and reports on probability theory, such as Heckerman [13], and we only present the results here. If the problem is a binomial one, in which only two mutually exclusive states exist, the Beta distribution is used. We discuss the Beta distribution in Section 3.1.2. If the problem is a multinomial one, in which r (r>2) mutually exclusive states exist, the Dirichlet distribution is used. We discuss the Dirichlet distribution in Section 3.1.3.

We use a car wash problem to illustrate the Bayesian approach of learning with data. If we have the car washed by the selected car wash, the outcome will be either "good" or "bad". Suppose we have the car washed n times, from the n observations, we want to estimate the probability of outcome "good" on the n+1th car wash.

In the classical analysis of this problem, we[5] assert that there is some physical probability of outcome "good", which is unknown. This classical probability is the physical property of the world (e.g., the probability that the outcome will be "good") [13]. A commonly-used estimator of classical probability is the maximum-likelihood (ML) estimator, which selects the probability θ using the following equation:

$$\theta_{ML}=g/n \tag{1}$$

where g represents the number of outcome "good" in the n observations. θ_{ML} then is used to estimate the probability of outcome "good" on the n+1th car wash [13].

In the Bayesian approach, we[6] also assert that there is some physical probability of outcome "good", but we encode our uncertainty about this physical probability using (Bayesian) probabilities, and use the rules of probability to compute the probability of outcome "good" on the n+1th car wash [13]. A Bayesian probability is a property of the person who assigns the probability (e.g., your degree of belief that the outcome will be "good"). In general, the process of measuring a degree of belief is commonly referred to as a probability assessment [13]. Unlike the classical probability, the Bayesian probability is subjective. We will show how to calculate the Bayesian probability in the following sections.

3.1.2 Beta distribution

In this section, we explain the Bayesian analysis of this car wash problem. It is a binomial problem with two states: "good" and "bad". We denote a variable by an upper-

[5] We refer to these analysts as frequentists

[6] We refer to these analysts as subjectivists

case letter (e.g. X, Y, X_i, Θ), and the state of a corresponding variable by that same letter in lower case (x, y, x_i, θ). We use $p(X=x|\xi)$ (or $p(x|\xi)$ as a shorthand) to denote the probability that $X=x$ for a person[7] with state of information ξ. We also use $p(x|\xi)$ to denote the probability distribution for X [13].

We define Θ to be a variable[8] whose values θ correspond to the possible true values of the physical probability [13]. We refer to θ as a parameter. We express the uncertainty about Θ using the probability density function $p(\theta|\xi)$ which represents the probability that Θ has the value θ given ξ . In addition, we use X_i to denote the variable representing the outcome of the i-th observation, i=1,…,n+1, and $D=\{X_1=x_1, \ldots, X_n=x_n\}$ to denote the set of observations. Thus, in Bayesian terms, the car wash problem reduces to computing $p(x_{n+1}|D, \xi)$ from $p(\theta|\xi)$ [13].

Returning to the car wash problem, we[9] use Bayes' rule[10] to obtain the probability distribution for Θ given observation D and background knowledge ξ [13]:

$$p(\theta \mid D, \xi) = \frac{p(\theta \mid \xi) p(D \mid \theta, \xi)}{p(D \mid \xi)} \tag{2}$$

where

$$p(D|\xi) = \int p(D|\theta,\xi) p(\theta|\xi) d\theta. \tag{3}$$

Next, we expand the term $p(D|\theta,\xi)$. Both Bayesians and classical statisticians agree on

---

[7] Note Bayesian probability is subjective.

[8] In Heckerman [13], "Bayesians typically refer to Θ as an *uncertain variable*, because the value of Θ is uncertain. In contrast, classical statisticians often refer to Θ as a *random variable*. In this section, we refer to Θ and all uncertain/random variables simply as variables."

[9] Actually the probability belongs to the person with state of information ξ. For simplicity, we introduce Bayesian estimation in the first person.

[10] A theorem in probability theory named for Thomas Bayes (1702-1761). According to wikipedia, Bayes' theorem (Bayes' rule) is a relation among conditional and marginal probabilities. It can be viewed as a means of incorporating informationm, from an observation, for example, to produce a modified or updated probability distribution. It can be restated as $P(A \mid B) = \frac{P(B \mid A) P(A)}{P(B)}$. The term P(A) is called the prior probability of A. P(A) also is the marginal probability of A. The term P(A|B) is called the posterior probability of A, given B. The term P(B|A), for a specific value of B, is called the likelihood function for A given B and can also be written as L(A|B). P(B) is the marginal probability of B.

this term: it is the likelihood function for binomial sampling [13]. In particular, given the value of Θ, the observation X_i in D are mutually independent, and the probability of outcome "good" on any one observation is θ (the probability of outcome "bad" is 1 - θ). Therefore, the likelihood function $p(D|\theta,\xi)$ is given by

$$p(D|\theta,\xi)= \theta^g(1- \theta)^b \tag{4}$$

where g and b are the number of outcome "good" and "bad" observed in observation D, respectively. Consequently, Equation 2 becomes [13]:

$$p(\theta \mid D,\xi) = \frac{p(\theta \mid \xi)\theta^g (1-\theta)^b}{p(D \mid \xi)} \tag{5}$$

The probability distributions $p(\theta|\xi)$ and $p(\theta|D,\xi)$ are commonly referred to as the *prior* and *posterior* for Θ, respectively. The quantities g and b are said to be *sufficient statistics*[11] for binomial sampling, because they provide a summarization of the data that is sufficient to compute the *posterior* from the *prior* [13]. Finally, we average over the possible values of Θ (using the mathematical expectation of θ) to determine the probability that the outcome of the n+1th car wash will be "good" [13]:

$$p(X_{n+1}=\text{“}good\text{”}\mid D,\xi)= \int p(X_{n+1}=\text{“}good\text{”}|\theta,\xi)p(\theta|D,\xi)d\theta \tag{6}$$

$$= \int \theta p(\theta|D,\xi)d\theta \equiv E_{p(\theta|D,\xi)}(\theta)$$

where $E_{p(\theta|D,\xi)}(\theta)$ denotes the expectation of θ with respect to the distribution $p(\theta|D,\xi)$.

In order to get the *posterior* distribution $p(\theta|D,\xi)$, we need a method to assess the *prior* distribution $p(\theta|\xi)$ for Θ. A common approach, usually adopted for convenience, is to assume that this distribution is a *beta* distribution [13]:

[11] In Benjamin Yakir [7], "let X denote a random variable (or vector) whose distribution depends on a parameter $\theta \in \Theta$. A real valued (or vector valued) function T of X is said to be sufficient if the conditional distribution of X, given T=t, is independent of θ (a.s., for all $\theta \in \Theta$)." According to About [2], the definition is as follows: "Suppose one has samples from a distribution, does not know exactly what that distribution is, but does know that it comes from a certain set of distributions that is determined partly or wholly by a certain parameter, q. A statistic is sufficient for inference about q if and only if the values of any sample from that distribution give no more information about q than does the value of the statistic on that sample. E.g. if we know that a distribution is normal with variance 1 but has an unknown mean, the sample average is a sufficient statistic for the mean."

$$p(\theta \mid \xi) = Beta(\theta \mid \alpha_g, \alpha_b) \equiv \frac{\Gamma(\alpha)}{\Gamma(\alpha_g)\Gamma(\alpha_b)} \theta^{\alpha_g - 1} (1-\theta)^{\alpha_b - 1} \qquad (7)$$

where $\alpha_g > 0$ and $\alpha_b > 0$ are the parameters of the *beta* distribution, $\alpha = \alpha_g + \alpha_b$, and $\Gamma(.)$ is the *Gamma* function[12]. The quantities α_g and α_b are referred to as *hyperparameters* to distinguish them from the parameter θ [13]. The quantities α_g and α_b determine the strength of the *prior* belief [33]. We discuss the strength of the *prior* belief in Section 4.3.4. By Equation 5, the *posterior* distribution will also be a *beta* distribution [13]:

$$p(\theta \mid D, \xi) = Beta(\theta \mid \alpha_g + g, \alpha_b + b) \equiv \frac{\Gamma(\alpha + n)}{\Gamma(\alpha_g + g)\Gamma(\alpha_b + b)} \theta^{\alpha_g + g - 1} (1-\theta)^{\alpha_b + b - 1} \qquad (8)$$

We say that the set of *beta* distributions is a conjugate family of distributions for binomial sampling. Also, the expectation of θ with respect to this distribution has a simple form [13]:

$$E_{p(\theta|D,\xi)}(\theta) = \int \theta Beta(\theta|\alpha_g+g, \alpha_b+b) d\theta = (\alpha_g+g)/(\alpha+n) \qquad (9)$$

Hence, given a beta *prior*, we have a simple expression for the probability of outcome “good” on the *n*+1th car wash [13]:

$$p(X_{n+1} = "good" \mid D, \xi) = E_{p(\theta \mid D, \xi)}(\theta) = \frac{\alpha_g + g}{\alpha + n} \qquad (10)$$

Assuming $p(\theta|\xi)$ is a *beta* distribution, it can be assessed in a number of ways, among which Heckerman [13] introduces two assessment methods: *imagined future data* and *equivalent samples*. In this thesis, the prior is tightly connected with the initial situational trust, which can be mapped from general trust or basic trust. We discuss how to assess the prior from the initial situational trust in Section 4.3.4.

We give an example to illustrate the Bayesian estimation. Suppose that 7 outcomes “good” have occurred in 10 previous car washes, Alice wants to estimate the probability of outcome “good” on the 11th car wash. Suppose that Alice’s prior belief in outcome “good” occurring in each car wash is *Beta*(1,1). Alice has the following estimation

$$P(X_{11} = "good") = \frac{1+7}{2+10} = \frac{2}{3} \approx 67\% \qquad (11)$$

[12] $\Gamma(x) = \int_0^\infty t^{x-1} e^{-t} dt$. If x is an integer n=1,2,3,…, then $\Gamma(n)=(n-1)!$.

In closing this section, we give several examples of *beta* distribution in Figure 1 (refer to Appendix D.1 for MATLAB code). To illustrated the result of the Bayesian estimation $p(X_{n+1}=\text{“}good\text{”}|D,\xi)= (\alpha_g+g)/(\alpha+n)$, we give the parametric mesh in Figure 2 (refer to Appendix D.2 for MATLAB code).

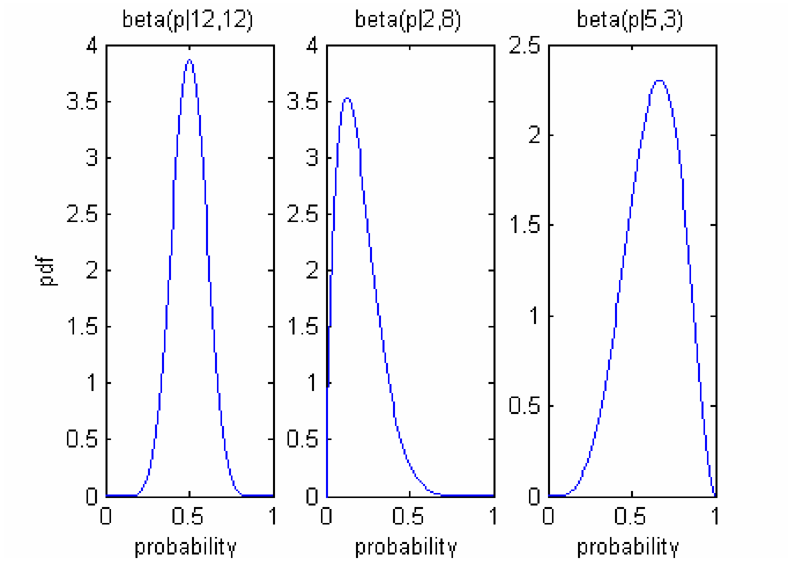

Figure 1: Beta distributions

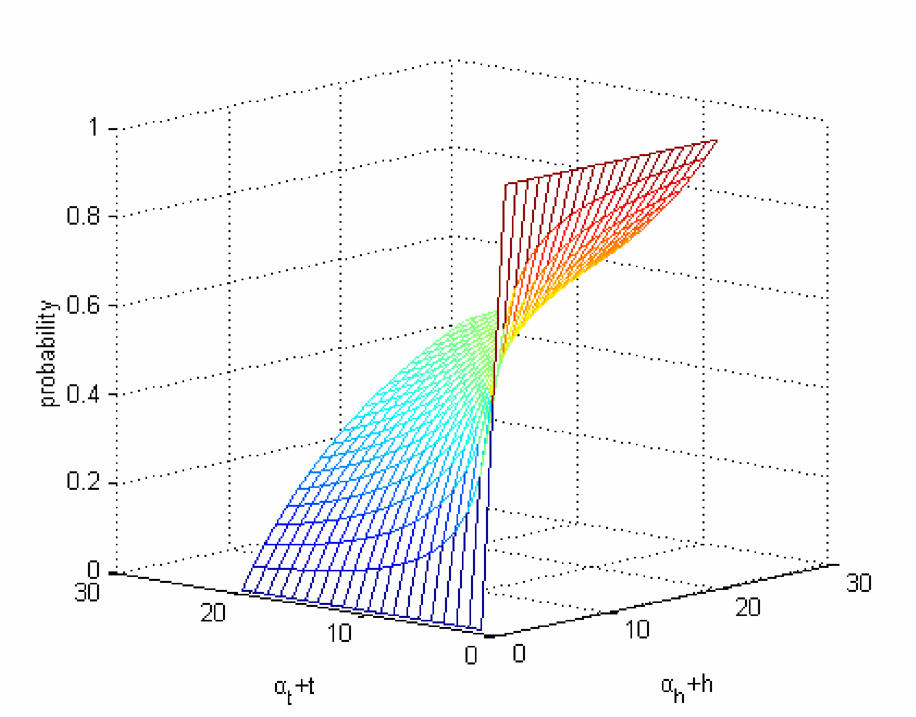

Figure 2: Bayesian estimation

3.1.3 Dirichlet distribution

In multinomial sampling, the observed variable X is discrete, having r (r>2) possible states $x^1,...,x^r$. The likelihood function is given by [13]

$$p(X=x^k|\boldsymbol{\theta},\xi)=\theta_k \quad k=1,...,r \tag{12}$$

where $\boldsymbol{\theta}=\{\theta_1, \theta_2,..., \theta_r\}$ are the parameters (the parameter θ_1, can be given by 1- θ_2 …- θ_r). The parameters correspond to physical probabilities of $x^1,...,x^r$. The *sufficient statistics* for observation data set $D=\{X_1=x_1, ...,X_n=x_n\}$ are $N=\{n_1,...n_r\}$, where n_k (k=1,…,r) is the number of times $X=x^k$ in D which consists of n independent identically distributed random outcomes. Here ξ is the background knowledge [13]. In Heckerman [13], the simple conjugate *prior* used with multinomial sampling is the Dirichlet distribution[13]:

$$p(\theta\mid\xi)=Dir(\theta\mid\alpha_1,...,\alpha_r)\equiv\frac{\Gamma(\alpha)}{\prod_{k=1}^{r}\Gamma(\alpha_k)}\prod_{k=1}^{r}\theta^{\alpha_k-1} \tag{13}$$

where α = $\sum\alpha_k$ and $\alpha_k>0,\ k=1,...,r$ and $\Gamma(.)$ is the *Gamma* function. The quantities $\alpha_1,…,\alpha_r$ are referred to as *hyperparameters* to distinguish them from the parameter $\boldsymbol{\theta}=\{\theta_1, \theta_2,..., \theta_r\}$. Following the similar process in the previous section, the *posterior* distribution is $p(\boldsymbol{\theta}|D,\xi)=Dir(\boldsymbol{\theta}|\alpha_1+n_1,...,\alpha_r+n_r)$. Techniques for assessing the prior beta distribution, including the methods of *imagined future data* and *equivalent samples*, can also be used to assess prior Dirichlet distributions [13]. In this thesis, the prior is tightly connected with the initial situational trust, which can be mapped from general trust or basic trust. We discuss how to assess the prior distribution from the initial situational trust in Section 4.3.4. Given this conjugate *prior* and data set D, the probability distribution for the next observation is given by

$$\begin{aligned} p(X_{n+1}=x^k|D,\xi) &= E_{p(\boldsymbol{\theta}|D,\xi)}(\theta_k) = \int\theta_k Dir(\boldsymbol{\theta}|\alpha_1+n_1,...,\alpha_r+n_r)d\boldsymbol{\theta} \\ &= (\alpha_k+n_k)/(\alpha+n). \end{aligned} \tag{14}$$

Another important quantity in Bayesian analysis is the *marginal likelihood* or *evidence* $p(D|\xi)$. In this case, we have [13]

$$p(D\mid\xi)=\frac{\Gamma(\alpha)}{\Gamma(\alpha+n)}\prod_{k=1}^{r}\frac{\Gamma(\alpha_k+n_k)}{\Gamma(\alpha_k)} \tag{15}$$

[13] The Beta distribution is a special case of the Dirichlet distribution in binomial sampling (r=2).

We note that the explicit mention of the state of knowledge ξ is useful, because it reinforces the notion that probabilities are subjective [13]. In the case r=2, the above analysis becomes equivalent to the case discussed in Section 3.1.2.

3.2 The Weighted Majority Algorithm

3.2.1 Introduction

The Weighted Majority Algorithm [27] studies the construction of prediction algorithms in a situation in which a learner faces a sequence of trials, with a prediction to be made in each, and the goal of the learner is to make few mistakes. The Weighted Majority Algorithm is interested in the case that the learner has reason to believe that one of some pool of known algorithms will perform well, but the learner does not know which one [27].

The Weighted Majority Algorithm (WMA) [27] deals with on-line prediction algorithms that learn according to the following protocol. Learning proceeds in a sequence of trials. In each trial, each algorithm of the pool makes a prediction and these predictions are fed to the *master algorithm*. The *master algorithm* then makes its prediction and receives a correct label, which it passes to the whole pool. In the update step, each algorithm's weight is multiplied by some factor that depends on its prediction in this trial [27].

WMA aims to design a *master algorithm* that uses the predictions of the pool to make its own prediction. Ideally the *master algorithm* should make not many more mistakes than the best algorithm of the pool, even though it does not have any a priori knowledge as to which of the algorithms of the pool make few mistakes for a given sequence of trials [27].

Littlestone and Warmuth [27] give the following definition of WMA:

Weighted Majority Algorithm (WMA): *Initially a positive weight is associated with each algorithm (function) of the pool. (All weights are initially one unless specified otherwise.) Algorithm WMA forms its prediction by comparing the total weight q0 of the algorithms of the pool that predict 0 to the total weight q1 of the* *algorithms predicting 1. WMA predicts according to the larger total (arbitrarily in case of a tie). When WMA makes a mistake, the weights of those algorithms of the pool that disagreed with the label are each multiplied by a fixed β such that $0 \le \beta < 1$.*

Note that the predictions of the algorithms in the pool and the master algorithm are all binary (that is, in {0,1}). If WMA is applied to a pool of functions with β=0 and the initial weights equal, then it is identical to the Halving Algorithm [27]. If β>0, the WMA gradually decreases the influence of functions that make a large number of mistakes and gives the functions that make few mistakes high relative weights. No single mistake can eliminate a function [27]. In this thesis, we assume that sequences of trials, the pool of functions, and labels are all finite.

Littlestone and Warmuth [27] proved that after each trial in which a mistake occurs the sum of the weights is at most u times the sum of the weights before the trial, for some u<1. In trials where no mistake occurs, the total weight may only decrease. Thus $w_{init} \times u^m \ge w_{fin}$ must hold, where m is the number of mistakes of WMA. This implies that m is at most $\log \frac{w_{init}}{w_{fin}} / \log \frac{1}{u}$. In the proof, Littlestone and Warmuth [27] assume $u=(1+\beta)/2$. Note that w_{init} is the total initial weight of all algorithms in the pool and w_{fin} is the total final weight after the given sequence of trails[14].

From the above results, Littlestone and Warmuth [27] give the following corollary:

Corollary 1: *Assume that A is a pool of n prediction algorithms and that m_i is the number of mistakes made by the i-th algorithm of the pool on a sequence S of instances with binary labels. Then WMA when applied to pool A with equal initial weights makes at most* $\dfrac{\log n + m_i \log \frac{1}{\beta}}{\log \frac{2}{1+\beta}}$ *mistakes on the sequence S, for* $1 \le i \le |A|$.

For example, suppose that there is a pool of 6 prediction algorithms and that the best algorithm makes m_i=20 mistakes on a sequence of 80 trials. Then by using corollary 1, we can estimate the upper bound of mistakes of WMA as follows (β=0.5):

[14] The choice of base is not significant for logarithms in this section. For formulas consisting of ratios of logarithms, we assume that the same base is chosen for numerator and denominator [27].

$$\text{the upper bound of mistakes of WMA} \le \frac{\log n + m_i \log\frac{1}{\beta}}{\log\frac{2}{1+\beta}} \approx 54\,.$$

Furthermore, we plot the relationship between the upper bound of mistakes made by the master algorithm and the parameter β in Figure 3 (refer to Appendix D.3 for MATLAB code).

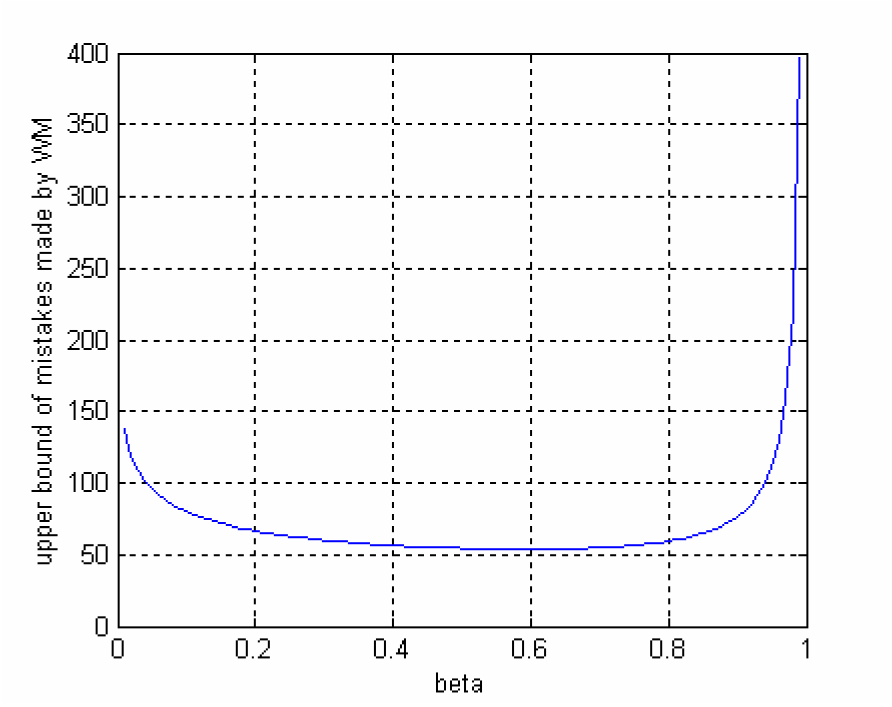

Figure 3: The upper bound of mistakes

3.2.2 WMA Continuous

There is a variant of WMA called WMA Continuous (WMC). WMC allows the predictions of the algorithms of the pool and the *master* as well as the labels to be in [0,1], instead of being binary. WMC simply predicts with the weighted average of the predictions of the pool algorithms [27].

The term update-trial j refers to the j-th trial in which an update step occurs [27]. We assume that there are a total of t such trials. The *master algorithm* is applied to a pool of n algorithms, letting $x_i^{(j)}$ denote the prediction of the i-th algorithm of the pool in update-trial j. Let $\lambda^{(j)}$ denote the prediction of the *master algorithm* in update-trial j, $\rho^{(j)}$ denote the label of update-trial j, and $w_1^{(j)},\dots, w_n^{(j)}$ denote the weights at the beginning of update-trial j. Consequently, $w_1^{(t+1)},\dots, w_n^{(t+1)}$ denote the weights following the final trial. All initial weights $w_1^{(1)},\dots, w_n^{(1)}$ are positive. The prediction of the *master algorithm* is

$$\lambda^{(j)} = \sum\nolimits^{n}_{i=1} x_i^{(j)} \times w_i^{(j)}/s^{(j)} \quad \text{where } s^{(j)}=\sum\nolimits^{n}_{i=1} w_i^{(j)}. \tag{16}$$

The prediction $x_i^{(j)}$ is multiplied by its relative weight $w_i^{(j)}/s^{(j)}$, which represents the "belief" of the *master algorithm* in the prediction. We have $s^{(1)}=w_{init}$ and $s^{(t+1)}=w_{fin}$.

In WMC, the notion of mistake has to be replaced by a quantity that measures how far the prediction is from the correct label. In this thesis, we use the absolute loss [27]. If an algorithm predicts x in a trial with label ρ, then its loss in that trial is $|x-\rho|$; this definition applies both to algorithms in the pool and to the *master algorithm*.

In an update step of WMC each weight $w_i^{(j)}$ is multiplied by some factor F that depends on β, $x_i^{(j)}$, and $\rho^{(j)}$: $w_i^{(j+1)}=F\times w_i^{(j)}$, where F can be any factor that satisfies

$$\beta^{|x_i^{(j)} - \rho^{(j)}|} \le F \le 1-(1-\beta)\,|x_i^{(j)} - \rho^{(j)}|. \tag{17}$$

The parameter β is the factor by which weights are multiplied and is always in the range [0,1). The parameter β measures how drastic the update is (the smaller β, the more drastic the update). For simplicity, we keep β constant for all trials [27].

Littlestone and Warmuth [27] give the following lemma [27]:

Assume that $w_i^{(i)}>0$ for $i=1,\dots,n$. Assume $0\le\beta<1$, $0\le\rho^{(j)}\le1$, and $0\le x_i^{(j)}\le1$ for $j=1,\dots,t$ and $i=1,\dots,n$. Assume $w_i^{(j+1)}\le w_i^{(j)}(1-(1-\beta)|\, x_i^{(j)}- \rho^{(j)}|)$ for $j=1,\dots,t$ and $i=1,\dots,n$. Then if $\beta=0$ and $|\lambda^{(j)}- \rho^{(j)}|=0$ for some j in $\{1,\dots,t\}$ then $w_{fin}=0$. Otherwise

$$\ln\frac{w_{fin}}{w_{init}} \le \sum_{j=1}^{t} \ln(1-(1-\beta)\,|\lambda^{(j)} - \rho^{(j)}|) \tag{18}$$

3.2.3 Similarity and distance

For the binomial case, the prediction and label are scalars. The absolute loss in WMC is $|x-\rho|$. Because the sum of the probabilities of all mutually exclusive and collectively exhaustive events is one, we can assign the probability $(1-x)$ to the negative event in the binomial case if the probability of the event is x. Therefore, for the scalar x, we can find a corresponding 2-D vector ***x***={x,1-x}, and for the scalar ρ, we can find a corresponding 2-D vector ***ρ***={ρ,1-ρ}. We now rewrite the absolute loss by using these two vectors as follows:

$$\text{Absolute loss} = |x-\rho| = \frac{|\mathbf{x}-\boldsymbol{\rho}|}{\sqrt{2}} \tag{19}$$

where |***x***- ***ρ***| is the Euclidean distance.

For the multinomial case, the prediction and label are multi-dimensional vectors. Intuitively, we use normalized Euclidean distance in multi-dimensional space to represent the absolute loss. Suppose prediction ***x*** and label ***ρ*** are r-dimensional vectors. The normalized Euclidean distance between ***x*** and ***ρ*** is

$$\text{Absolute loss} = \frac{|\mathbf{x}\text{-}\boldsymbol{\rho}|}{\sqrt{2}} = \frac{\sqrt{\sum_{i=1}^{r}(x_i-\rho_i)^2}}{\sqrt{2}}. \tag{20}$$

Therefore the absolute loss is in the range [0,1], whatever the dimension of predictions and labels is. Please refer to Appendix A for details about the range of distance.

There are several other possible distance metrics (e.g. Minkowski metrics, Clark's distance, Cosine distance). We explain some of them in Appendix A.

4 A trust model with statistical foundation

In this chapter we define a trust model with statistical foundation.. Such a foundation is intuitive and useful in many practical situations, as will be shown in Section 4.5 on “Decision making”. In this chapter, we focus on how to build trust from self experience.

4.1 Trust representation

One of the key issues for the design of trust model is how trust is represented, and how the effect of experiences is specified. Representations can be qualitative, using specific qualitative labels (or term structures), or quantitative, using numbers as a representation [11].

Qualitative trust representation is just sufficiently rich to specify a difference in characteristics between slow and fast dynamics, but it is not rich enough to specify more subtle differences in characteristics [11]. Such a representation causes the loss of sensitivity and accuracy [36]. Once an entity is judged to be “Trustworthy”, then another entity that is more trustworthy, but not enough to be “Very Trustworthy”, suffers in the comparison [36].

In quantitative representation, trust is measured by a real value, which is bounded between lower and upper limits. Marsh [36] chooses to represent trust as a continuous variable over range [-1, 1). The value -1 is the lowest possible trust value: it represents blind distrust. The value +1 means blind trust. Any real number between -1 and 1 means conditional trust. The higher the trust value, the more trustworthy the entity is. However, entity-subjectivity in the use of values is a major problem. We face the problem “what does a trust of 0.5, or 50% mean? It is high or low”. Such a representation of trust introduces ambiguity into the model, as the semantics of trust are usually hard to represent as a single real value [3].

Trust is a very complex and multi-faceted thing, which is hard to represent as either a single quantitative value or some qualitative rating. In either qualitative or quantitative representation, trust is a scalar, which we believe could cause potential information loss [36] or ambiguity [3]. In this thesis, we propose that trust is the estimation of a probability distribution over possible outcomes of experiences. The space of possible outcomes usually depends on the context in which the trust model is used. This approach provides a statistical foundation to the trust model and allows its application to a variety of different usage scenarios.

4.2 A model of the trusted entity

Our trust model is based on a model of the trusted entity β[15]. We discuss the space of possible outcomes with respect to a service performed by β and then propose a stochastic model for β.

[15] From the point view of the trusting entity α.

4.2.1 The space of possible outcomes

Our trust model is based on an abstract model of the trusted entity. We assume that the trust concerns the execution of a certain action by the entity. In most cases, the execution of the action corresponds to a specific service that is provided by the trusted entity. There may be different outcomes of the action. The trust is concerned with some form of prediction of what the outcome will probably be. In the case of situational trust, we are concerned with a particular action in a certain situation; in the case of general trust, the action represents any action of the trusted entity that may be of interest [19].

It is important to identify the space of possible outcomes. This space determines the nature of the associated trust model. We note that the granularity of this space determines the precision with which any prediction of future behavior can be made. We give in the following some typical examples.

(a) Discrete categories [19]

In this case, the outcomes are classified into a finite set of categories. For instance, the eBay trust model foresees the three categories: “positive”, “neutral”, and “negative”. In the case of trust concerning the quality of the food in a restaurant, the categories may be “excellent”, “good”, “average”, “bad”, and “very bad”. The case of two-valued outcomes is a special case of discrete categories; here the outcomes are classified into two categories, which may be called “good” and “bad”.

While in the above examples, the different categories were ordered according to some intuitive “goodness” relationship (“good” being better than “average”, for instance), there are cases in which such an ordering does not necessarily exist. We may consider the example where the outcomes are classified into the following categories: “normal: all options OK”, “option A failed”, and “option B failed”. Here it is not clear which of the last two categories would be better.

(b) Numeric outcomes [19]

There are many cases in which the outcome can be characterized by a numerical value. For instance, the trust may concern the response time of a Web server, or the delivery delay of a parcel delivery service. In these cases, we are interested in the delay for completing the action, and this delay may be measured in fractions of seconds, minutes, or hours, depending on the precision that is reasonable for the application. In these cases, the number of different outcomes is in principle infinite.

Other examples where the outcomes can be classified by a numerical value are the following: (1) What percentage of cost overruns can one expect in a construction contract? – or (2) What is the expected quality of a video obtained from a video-on-demand service?

(c) Multidimensional outcome characterization [19]

In many situations, the outcome of the action of interest has several parameters that are important to consider. Each of these parameters can usually be characterized either by a value from discrete value space, or a numerical (integer or real) value. In this case, we say that the space of the possible outcomes is multi-dimensional (one dimension for each parameter). Here are two examples:

(1) Restaurant service with several evaluation criteria: (i) quality of food, (ii) service, and (iii) environment. For each of these three criteria, the restaurant may be classified into a certain number of discrete values, such as "excellent" down to "very bad". Therefore, the outcome of a restaurant experience may be classified as a point in this three-dimensional space, where each coordinate in this space is defined by a value between "excellent" and "very bad".

(2) Multimedia presentation quality: As explained in [5] and [1], the quality of a multimedia presentation may be characterized by three values: (i) frame rate (in video frames per second), (ii) resolution (number of pixels within a frame), and (iii) color quality (number of colors distinguished per pixel). Therefore, the outcome of a video presentation obtained from a video-on-demand service may be characterized by three numerical values corresponding to these three quality of service parameters.

4.2.2 A stochastic model of the trusted entity

We assume that the trusted entity behaves like a stochastic process, in the sense that the outcome of an action of interest cannot be predicted exactly, that the outcome of one execution of an action of interest is statistically independent of the outcome of previous executions of that action, and that, over the long run, the probability that the outcome for the next execution of the action will be a particular point within the space of possible outcomes is described by a probability distribution, which we call the **outcome distribution** of the trusted entity, and which we represent by D_β. The value of D_β for a particular outcome $x \in X$ (where X is the space of possible outcomes) is written as $D_\beta(x)$. The outcome distribution is a distribution over the space of possible outcomes. Therefore the sum over all possible outcomes of the outcome distribution must be equal to one.

In the case of discrete outcome spaces, one usually does not make any assumptions about relationships between the outcome probabilities for different outcomes (except that they must sum to one). However, in the case of numerical outcomes, one may introduce additional assumptions. For instance, in Figure 4, a Gaussian outcome distribution is assumed, and the parameters of the Gaussian distribution are determined from a histogram of the outcomes observed during multiple experiments [19].

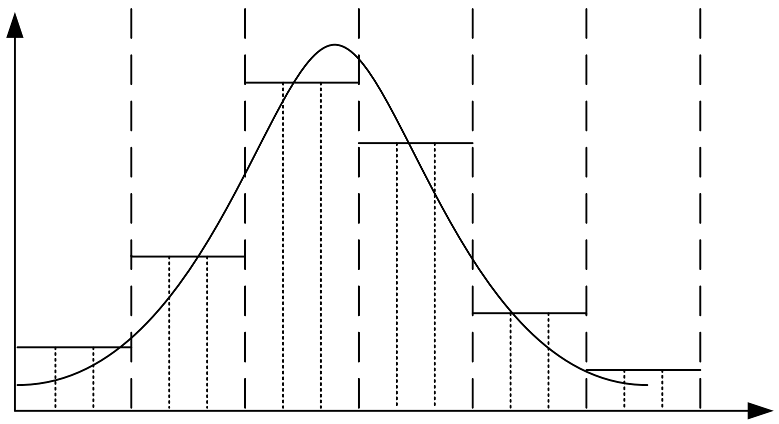

Figure 4: The Gaussian outcome distribution [19]

4.3 Building trust from self experience

We now define trust and propose a model to build trust from prior experiences. For this section, a formal analysis of the dependency of trust on self-experiences will be the central focus. In this section, we explain how to apply Bayesian estimation to our trust model. Our approach involves two major steps: (1) setting initial situational trust and *prior* hyperparameters; (2) building trust from self experience.

4.3.1 Definition of trust

Definition of trust: The trust of an entity *α* in the outcome of an action of entity *β* is an estimation of the outcome distribution D_β for the execution of the action by entity *β*.

The basic mechanism for building trust is by experience, that is, by observing the execution of the action of interest by the entity *β* a certain number of times. The entity predicts the distribution of the next observation based on self experience using the Bayesian estimation. The Bayesian estimation and the mathematical analysis leading to the expression for *posterior* distribution can be found in Chapter 3, and we only present the results here. Let us assume that the space of possible outcomes *X* is discrete and finite and that *n* observations have been made, where the outcome of the *i*-th observation was X_i.

We assume that the observed variable *X* is discrete; having *r* ($r\geq 2$) possible states $x^1,\ldots,x^r$. The simple conjugate *prior* $p(\boldsymbol{\theta}|\xi)$ used with multinomial sampling[16] is the Dirichlet distribution [13]:

$$p(\boldsymbol{\theta} \mid \xi) = Dir(\boldsymbol{\theta} \mid \alpha_1,\ldots,\alpha_r) \equiv \frac{\Gamma(\alpha)}{\prod_{k=1}^{r}\Gamma(\alpha_k)} \prod_{k=1}^{r} \theta_k^{\alpha_k - 1} \qquad \textbf{(21)}$$

where $\alpha = \sum\alpha_k$ and $\alpha_k>0$, $k=1,\ldots,r$. The quantities $\alpha_1,\ldots,\alpha_r$ are referred to as hyperparameters to distinguish them from the parameter $\boldsymbol{\theta}=\{\theta_1, \theta_2,\ldots, \theta_r\}$ (the parameter θ_1 can be given by $1-\theta_2\ldots-\theta_r$), which correspond to physical probabilities of $x^1,\ldots,x^r$. Suppose the trusting entity has experience denoted by an observation data set $D=\{X_1=x_1,\ldots,X_n=x_n\}$ for which the *sufficient statistics* are $N=\{n_1,\ldots,n_r\}$, where n_k is the number of times $X=x^k$ in *D* which consists of *n* independent identically distributed random outcomes. Here ξ is the background knowledge[17]. Given conjugate *prior* $p(\boldsymbol{\theta}|\xi)$ and data set *D*, the *posterior* distribution $p(\boldsymbol{\theta}|D, \xi)$ and the probability distribution for the next observation $p(X_{n+1}=x^k|D, \xi)$ are given by

[16] The Beta distribution is a special case of the Dirichlet distribution in binomial sampling. Therefore we only use the Dirichlet distribution in this section.

[17] The background belongs to the trusting entity (entity *α*). The prior and posterior distributions also belong to entity *α*.

$$p(\boldsymbol{\theta}|D, \xi)=Dir(\boldsymbol{\theta}|\alpha_1+n_1,\dots, \alpha_r+n_r) \quad \textbf{(22)}$$

$$p(X_{n+1}=x^k|D,\xi)=E_{p(\boldsymbol{\theta}|D,\xi)}(\theta_k)=\int\theta_k Dir(\boldsymbol{\theta}|\alpha_1+n_1,\dots,\alpha_r+n_r)d\boldsymbol{\theta}=(\alpha_k+n_k)/(\alpha+n) \quad \textbf{(23)}$$

Then the estimation of D_β, the trust of the observing entity α, is given by the formula

$$T_\alpha(\beta,\delta)(x^k) = p(X_{n+1}=x^k|D,\xi) = (\alpha_k+n_k)/(\alpha+n). \quad (24)$$

We note that the explicit mention of the state of knowledge ξ is useful, because it reinforces the notion that probabilities are subjective. When set r=2, the results are suitable for binomial sampling.

For simplicity, we sometimes use $E_{p(\boldsymbol{\theta}|D,\, \xi)}(\boldsymbol{\theta})$ (or *$E(\boldsymbol{\theta})$ for short)* to denote situational trust $T_\alpha(\beta,\delta)(x)$. Note that $E_{p(\boldsymbol{\theta}|D,\, \xi)}(\boldsymbol{\theta})$ is an r-dimensional vector. For each x^k, θ_k, we have $T_\alpha(\beta,\delta)(x^k) = E(\theta_k)$ *where* k=1,…,r.

In the case that the space of possible outcomes includes a dimension with a numerical coordinate, the set of possible outcomes becomes infinite. In this case, the above simple average value calculation is not possible. Instead, the numerical coordinate is usually partitioned into a discrete number of intervals, as shown in Figure 4. Each interval is then treated like a discrete value and the above formula can be applied. If the model of the trusted entity includes an assumption about the functional form of the outcome distribution function D_β then the trust should be of the same form, and the parameters of this function should be adjusted to best fit the experimental data [19].

Instead of keeping in memory all previous experimental outcomes, the trusting entity may use *sufficient statistics* for the previous outcomes. For calculating the trust, the trusting entity keeps in memory the current *sufficient statistics* $N=\{n_1,\dots,n_r\}$. When a new experience yielding outcome x^k is observed, the *sufficient statistics N* will be updated as follows:

$$n_k = n_k+1 \quad (25)$$
$$n=n+1$$

The value of $T_\alpha(\beta,\delta)$ can be calculated as follows:

$$T_\alpha (\beta,\delta) (x^i) = (\alpha_i+n_i)/(\alpha+n) \;\; for\ i=1,\dots,r \quad (26)$$

In our trust model entities do not maintain a database of specific trust statements in the form of "α trusts β with respect to situation δ". Instead, at any given time, the trustworthiness of a particular entity is obtained by summarizing the relevant subset of recorded experiences [3].

4.3.2 Considering trusted entities with evolving performance

If it can be assumed that the performance of the trusted entity is not constant, but evolving over time, then the basic assumption about a given outcome distribution for the actions of the entity, valid over all times, is not true any more. In this case, we must take into account that the outcome distribution of the trusted entity evolves over time. If the trusting entity knows the speed of this evolution, possibly defined by a given characteristic time delay, then the trusting entity may include in the trust calculation only recent experiments not older than the characteristic time delay.

It is also possible to give different weights to the different experiments, either according to their age or their order. It is desirable to give greater weight to more recent experiences[18]. This can be achieved by introducing a forgetting factor γ where $0 \leq \gamma \leq 1$. When a new experience yielding outcome x^k is observed, the *sufficient statistics N* will be updated as follows:

$$n_k = n_k \times \gamma + 1 \qquad (27)$$

$$n_i = n_i \times \gamma \quad \textit{for } i=1,\ldots r \textit{ and } i \neq k$$

$$n = n \times \gamma + 1$$

where the value of γ determines the weight of the past experience compared with the most recent experience. We note the largest possible n is $1/(1-\gamma)$. Please refer to appendix C for details. The values of $T_\alpha(\beta,\delta)$ can be calculated as follows:

$$T_\alpha(\beta,\delta)(x^i) = (\alpha_i + n_i)/(\alpha + n) \quad \textit{for } i=1,\ldots,r \qquad (28)$$

4.3.3 States and dimension independence

In many situations, the space of the possible outcomes is multi-dimensional (one dimension for each parameter). Consider the restaurant service with three evaluation criteria: (i) quality of food, (ii) service, and (iii) environment. For each of these three

[18] It may be generally true, but in the context of trust not always. Entities may be as likely to give more weight to "significant" experiences. E.g. if I am risk averse I may put a very high value on bad experiences even if they have not happened for awhile.

criteria, the restaurant may be classified into four values, such as “excellent”, “good”, “bad”, and “very bad”. Therefore, the outcome of a restaurant experience may be classified as a point in this three-dimensional space, where each coordinate in this space is defined by a value between “excellent” and “very bad”. For example, one possible outcome (state) is *x=(“Quality of food”=“excellent”, “service”=“good”, “environment”=“bad”)*. The concept of the multidimensional space is useful because it makes the abstract outcome space easy for human to understand. However, the Bayesian estimation methods do not need such concept of multidimensional space. What Bayesian estimation needs is just the definition of this space of outcome states. The Bayesian estimation does not need to make any further assumptions about the components of this outcome x.

Besides the ease to understand, the concept of multidimensional space has another potential benefit: ease to compute. Following the restaurant example, we assume the three evaluation criteria are mutually independent; the number of outcome state for each dimension is four. In the case of the independent multidimensional outcome space, $T_\alpha(\beta,\delta)(x^k)=P(x^k)=\Pi^m{}_{i=1}P(o_i)\ k=1,\ldots,r$ where $P(x^k)$ is the probability of outcome x^k, $x^k=(o_1,o_2,\ldots,o_m)$ and o_i is the outcome in i-th dimension (in this case, each o_i has four possible states, m is equal to three). Here the marginal distribution of o_i can be used instead of the joint distribution of x^k because the dimensions are independent. The dimension independence can significantly reduce the computational complexity especially when the number of outcome states is large. For the example of the restaurant, the number of outcome states is r=4^3=64 and the dimension number is m=3. In some real situations, the number could be even larger. However, if we assume the evaluation criteria are mutually independent, the number of outcome states for each dimension is r=4, which is more manageable. For each dimension, the Bayesian estimation can be applied independently to get the marginal distributions $P(o_i)$. Then the joint distribution of x^k can be calculated from the product of the marginal distributions $P(o_i)$. We note that all the examples in this thesis assume the dimension independence without notice.

4.3.4 Setting initial situational trust and *prior* hyperparameters

The basic idea in this section is to let the conjugate *prior* match the initial situational trust, which can be set from general trust or basic trust. In the previous sections, we model trust $T_\alpha(\beta,\delta)(x)$ as the distribution $D_\beta(x)$ over the space of possible outcomes X. In Section 4.4, we will propose a mapping function $S(x)$ to set initial situational trust $T_\alpha(\beta,\delta)(S(x))$ from general trust $T_\alpha(\beta)(x)$ or basic trust $T_\alpha(x)$ with initial observation number N_{init}. In this section, we assume that the initial situational trust as well as the initial observation number is known. For simplicity without loss of generality, we suppose there are r ($r\geq 2$) different outcomes $x^1,\ldots, x^r$ in this situation δ. In the Bayesian technique, the likelihood function is given by

$$p(X=x^k|\boldsymbol{\theta}, \xi) = \theta_k,\ \ k=1,\ldots,r \qquad \textbf{(29)}$$

where $\boldsymbol{\theta}=\{\theta_1, \theta_2,\ldots, \theta_r\}$ are the parameters (the parameter θ_1 can be given by 1 $-\theta_2\ldots-\theta_r$) which correspond to physical probabilities of $x^1,\ldots, x^r$. The simple conjugate *prior* used with multinomial sampling is the Dirichlet distribution:

$$p(\boldsymbol{\theta} \mid \xi) = Dir(\boldsymbol{\theta} \mid \alpha_1,\ldots,\alpha_r) \equiv \frac{\Gamma(\alpha)}{\prod_{k=1}^{r}\Gamma(\alpha_k)} \prod_{k=1}^{r} \theta_k^{\alpha_k -1} \qquad \textbf{(30)}$$

where $\alpha = \sum\alpha_k = N_{init}$ and $\alpha_k>0$, k=1,…,r. The hyperparameters $\alpha_{1,\ldots,}\ \alpha_r$ can be assessed as follows.

Initial situational trust $T_\alpha(\beta,\delta)(x^k) = p(X=x^k|\boldsymbol{\theta}, \xi) = \theta_k= \alpha_k/\alpha\ \ k=1,\ldots,r.$ **(31)**

Given these r equations, we can solve for $\alpha_{1,\ldots,}\ \alpha_r$. The size of N_{init} determines the strength of the prior beliefs. If this N_{init} is small, such as 2, just a few observations will be enough to take over prior beliefs [33]. On the other hand, if the N_{init} is large, such as 1000, then on the order of 1000 observations will be needed to significantly make the *posterior* distribution differ from the prior beliefs [33]. When N_{init} approaches 0, the *prior* distribution becomes noninformative (theoretically N_{init} can be any positive real number).

4.3.5 Example

Following the restaurant example, we assume the evaluation criteria are mutually independent; the number of outcome states for each dimension is four. For each dimension, the Bayesian estimation is applied independently. The final joint distribution of x^k can be calculated from the product of the marginal distributions. We do not show the distribution of x^k because it is very easy to calculate and it occupies lots of space.

Suppose the trusting entity α has seven experiences ($n=7$) denoted by an observation data set $D=\{X_1=x_1,\ldots,X_7=x_7\}$. The *sufficient statistics* in each dimension for the data set are shown in Table 1.

	Quality of food	Service	Environment
excellent	5	2	3
good	1	3	3
bad	1	1	1
very bad	0	1	0

Table 1: Sufficient statistics in a restaurant based on seven experiences

We assume the prior belief (*hyperparameters*) in each dimension in restaurant β as shown in Table 2

	Quality of food	Service	Environment
excellent	1	0	1
good	1	1	0
bad	0	1	0
very bad	0	0	1

Table 2: Prior belief in a restaurant

At this point of time, Entity α can calculate the situational trust in each dimension in restaurant β as shown in Table 3

Distribution $T(o)$	Quality of food $T1(o)$	Service $T2(o)$	Environment $T3(o)$
excellent	6/9	2/9	4/9
good	2/9	4/9	3/9
bad	1/9	2/9	1/9
very bad	0	1/9	1/9

Table 3: Situational trust in a restaurant based on seven experiences

After entity α obtains another outcome such as X_8= *("Quality of food"= "excellent", "service"="good", "environment"="bad")*, the entity α updates his/her experience as shown in Table 4

	Quality of food	Service	Environment
excellent	6	2	3
good	1	4	3
bad	1	1	2
very bad	0	1	0

Table 4: Sufficient statistics in a restaurant based on eight experiences

With the updated experience, the entity *α* can calculate the new situational trust and obtain the following trust in each dimension (*n=8*) as shown in Table 5

Distribution *T(o)*	Quality of food *T1(o)*	Service *T2(o)*	Environment *T3(o)*
excellent	7/10	2/10	4/10
good	2/10	5/10	3/10
bad	1/10	2/10	2/10
very bad	0	1/10	1/10

Table 5: Situational trust in a restaurant based on eight experiences

4.4 Initial trust values

In two cases, entity *α* needs to set his/her initial trust values in entity *β*. (i) When entities *α* and *β* have no previous relationship (in any situation) and entity *α* has no knowledge about entity *β*, then entity *α* needs to initialize his/her general trust and situational trust in entity *β*. (ii) When entities *α* and *β* have no previous relationship in a new situation but entity *α* has general trust in entity *β*, then entity *α* needs to initialize his/her new situational trust in entity *β*. To address these problems, a mapping between different spaces is needed. Mapping to initial trust for a particular entity or situation depends on the space of possible outcomes of that situation.

4.4.1 Mapping between spaces

We focus on the following two mappings[19]:

(i) Generalization mapping: from situational trust space to general trust space for the purpose of general trust update. We write *G(x)* for the outcome of general trust when the situational trust outcome is *x*. Using *G(x)* one can update his/her general trust $T_\alpha(\beta)(G(x))$. Note that this kind of generalization mapping causes information loss since the general trust would be more "general" (abstract) in nature and the mapping is usually a many-to-one mapping, which implies that the number of discrete outcomes of general trust space must be no more than that of the situational trust space. An example of a mapping from the situational trust to the general trust is illustrated in the following Figure 5.

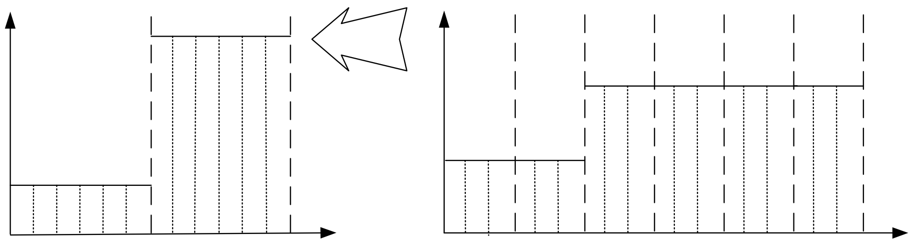

Figure 5: Generalization mapping

In this example, the entity α will map his/her situational trust to general trust by defining the mapping *G(x)*:

- outcomes "average" or higher in situational trust map to outcome "Good" in general trust
- outcomes "Bad" or lower in situational trust map to outcome "Bad" in general trust

Note that the areas must be the same; that is, $T_\alpha(\beta)(G(x)) = T_\alpha(\beta,\delta)(x)$. Thus

[19] It is also possible to map the *sufficient statistics* between different spaces.

$T_{\alpha}(\beta)(\text{"Good"}) = T_{\alpha}(\beta,\delta)(\text{"Excellent"}) + T_{\alpha}(\beta,\delta)(\text{"Very Good"}) + T_{\alpha}(\beta,\delta)(\text{"Good"}) + T_{\alpha}(\beta,\delta)(\text{"Average"}) = 80\%$, and

$T_{\alpha}(\beta)(\text{"Bad"}) = T_{\alpha}(\beta,\delta)(\text{"Bad"}) + T_{\alpha}(\beta,\delta)(\text{"Very Bad"}) = 20\%.$

(ii) Specialization mapping: from basic trust space to general trust space and from general trust space to situational trust space for the purpose of setting initial trust values. We write *S(x)* for the outcome of situational trust when the general trust outcome is *x*. We also write *S(x)* for the outcome of general trust when the basic trust outcome is *x*. Using *S(x)* one can set initial situational trust $T_{\alpha}(\beta,\delta)(S(x))$ and initial general trust $T_{\alpha}(\beta)(S(x))$. Note that the specialization mapping is the reverse process of the generalization mapping. It usually is a one-to-many mapping. An example of a mapping from the general trust to the situational trust is illustrated in the following Figure 6.

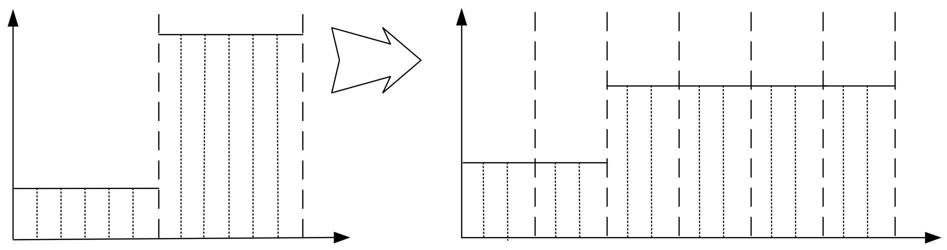

Figure 6: Specialization mapping

In this example, the entity *α* will map his/her general trust to situational trust by defining the mapping *S(x)*:

- outcome "Good" in general trust maps to outcomes "average" or higher in situational trust
- outcome "Bad" in general trust maps to outcomes "Bad" or lower in situational trust

Note that the areas must be the same; that is, $T_{\alpha}(\beta)(x) = T_{\alpha}(\beta,\delta)(S(x))$. Thus

$T_\alpha(\beta)(\text{“Good”}) = T_\alpha(\beta,\delta)(\text{“Excellent”}) + T_\alpha(\beta,\delta)(\text{“Very Good”}) + T_\alpha(\beta,\delta)(\text{“Good”}) + T_\alpha(\beta,\delta)(\text{“Average”}) = 80\%$, and

$T_\alpha(\beta)(\text{“Bad”}) = T_\alpha(\beta,\delta)(\text{“Bad”}) + T_\alpha(\beta,\delta)(\text{“Very Bad”}) = 20\%.$

This histogram is then the initial set of values for situational trust outcomes (i.e., $n = N_{init}$) that will be updated over time as entity α has further interactions with service β.

4.5 Decision making

Our goal is to develop a general trust model that can be used for making rational decisions in order to make optimal choices. In other words, we believe that we should use trust to support decision making. We highlight the decision making process because trust is a very complex and multi-faceted thing and it is generally very hard to make decisions directly from trust. We believe that we typically make decision based on utility we expect to gain. In management and marketing, many utility models have been developed, while little has been done in combining utility models and trust models. In this section, we are going to develop several utility models in different contexts to see how our trust model can apply to these utility models. For this section the decision-making is the focus.

In this section, we apply our trust model to several utility models to show how our trust model can be used for rational decision making. For most economic scenarios, the highest expected current utility model [20] is appropriate. For some critical scenarios, the lowest expected failure rate model [28] is appropriate. For some service scenarios, the total satisfaction model [5] is appropriate.

Based on our trust model and Expected Utility Theory (EUT), we propose the following: if entity α wants to use his/her trust for decision making, the entity should first establish the utility of the action of a trusted entity β for each possible outcome. We write $U_\alpha(x)$ for the utility when the outcome is x. Then it is clear that the expected utility obtained from the execution of an action by entity β for which the trust is $T_\alpha(\beta,\delta)$ can be calculated by the formula

$U_\alpha(\beta) = \sum_{x \in X} T_\alpha(\beta,\delta)(x) \times U_\alpha(x)$

In the case of multi-dimensional outcome spaces, the different dimensions may have their own utility mapping functions, and the overall utility may be the sum of the single-dimension utilities, adjusted with weight factors for the different dimensions. We then get an analogous formula to the one given in [20]. If all dimensional outcomes are independent, then the above expected utility formula can be generalized to $U_\alpha(\beta) = \sum_{k=1}^{K} U_\alpha^{(k)}(\beta) \times w_\alpha^{(k)}$ where $U_\alpha^{(k)}(\beta)$ is the expected *k*-th dimensional utility, $w_\alpha^{(k)}$ is the subjective weight of the *k*-th dimension (we assume that the sum of all weights is equal to 1).

We note that the latter formula corresponds to the formula for the expected utility quoted from [20] above. $U_\alpha^{(k)}(\beta)$ in our formula corresponds to the value y_{jk} in the formula above.

We give three examples of making decisions and choosing the utility mapping function $U_\alpha(x)$.

(1) Consider the example of the restaurant service β. Entity α assumes that all three evaluation criteria are independent. Let us assume that entity α adopts the mapping functions of Table 6 and dimensional weights[20] with the following values: $W^{(1)} = 0.6$; $W^{(2)} = 0.3$; $W^{(3)} = 0.1$.

Utility Mapping $U(o)$	Quality of food $U^{(1)}(o)$	Service $U^{(2)}(o)$	Environment $U^{(3)}(o)$
excellent	5.6	3	2
good	2.7	1	1
bad	0	-0.5	0
very bad	-4	-2	-1

Table 6: Utility mapping functions for the restaurant service

The weighted "quality of food" dimension utility can be calculated using the trust values from Table 5 as follows:

U1= sum over all *o* in dimension "quality of food" of ($U^{(1)}(o) * T1(o) * W^{(1)}$)

[20] Both the assignments of utility and dimensional weights are arbitrary in this example

= (5.6 * (7/10) + 2.7 * (2/10) + 0.0 * (1/10) + (-4) * 0) * 0.6 = 2.676

Similarly, the weighted “service” dimension utility *U2* has the value 0.24, and the weighted “environment” dimension utility *U3* has the value 0.1. Therefore the utility for entity *α* of this restaurant service *β* is $U_\alpha(\beta) = U1 + U2 + U3$ = 2.676 + 0.24 + 0.1 = 3.016

Following the same process, entity *α* can calculate the utility of other restaurant services. Entity *α* would choose the restaurant service with the highest utility value.

(2) Consider the example of multimedia presentations. Based on the multi-dimensional outcome space discussed at the end of Section 4.2.1, we could use the above formula to calculate the overall utility. However, Richards *et al.* [5] propose another formula. They call satisfaction s_k what we call utility $U^{(k)}$, and they assume that the values of satisfaction range between zero (unacceptable quality) and one (ideal quality). Instead of the weighted summation formula above, they propose to calculate the overall satisfaction by $S_{total} = K / \sum_{k=1}^{K} \frac{1}{s_k}$. The reason for proposing this formula is the following argument: If the satisfaction for one dimension is zero, then the total satisfaction should be zero (which is not satisfied by our formula). Both formulae satisfy the following property: If the satisfaction for all dimensions has the same value, then the overall satisfaction has that same value. Richards' formula can be extended to include weights.

Let us assume that entity *α* adopts the following satisfaction mapping functions in different dimensions as shown in Table 7, Table 8, and Table 9.

Frame rate(F)	F>25	20<F<25	10<F<20	F<10
Satisfaction(U1)	1	0.8	0.6	0.3

Table 7: Satisfaction in frame rate dimension

Resolution(R)	640x480	352x240	160x120
Satisfaction(U2)	1	0.8	0.7

Table 8: Satisfaction in resolution dimension

Color(C)	$C=2^{16}$	$C=2^8$
Satisfaction(U3)	1	0.5

Table 9: Satisfaction in color dimension

Let us assume that entity *α* adopts the following situational trust in each dimension in the video-on-demand service provider *β,* as shown in Table 10

Frame rate *T1(o)*		Resolution *T2(o)*		Color *T3(o)*	
F>30	2/10	640x480	7/10	$C=2^{16}$	4/10
20<F<30	5/10	352x240	2/10	$C=2^8$	6/10
10<F<20	2/10	160x120	1/10	N/A	N/A
F<10	1/10	N/A	N/A	N/A	N/A

Table 10: Situational trust in a video-on-demand service provider

The “frame rate” dimension satisfaction can be calculated using the trust values from Table 10 as follows:

S1= sum over all *o* in dimension “frame rate” of ($U^{(1)}(o) * T1(o)$)

= 1* (2/10) + 0.8 * (5/10) + 0.6 * (2/10) + 0.3 *(1/10) = 0.75

Similarly, the “resolution” dimension satisfaction *S2* has the value 0.93, and the “color” dimension satisfaction *S3* has the value 0.7. Therefore the total satisfaction for entity *α* of this video-on-demand service provider *β* is $S_\alpha(\beta) = 3/(1/S1 + 1/S2 + 1/S3) \approx$ 0.78.

Following the same process, entity *α* can calculate the utility of other video-on-demand services. Entity *α* would choose the video-on-demand service with the highest total satisfaction value.

(3) Consider the previous example of restaurant service *β*. Entity *α*, this time, uses a failure probability model similar to failure rate as proposed in [28] for decision making. Entity *α* first maps the outcome space to a consideration space which consists of two outcomes, namely “success” and “failure”; for instance, we may assume that we have "failure" when the value of $U_\alpha(x)$ is less than zero. The service failure probability is the proportion of outcome “failure” and can be represented by $P_f = \sum_{U_\alpha(x)="failure"} T_\alpha(\beta,\delta)(x)$. If the dimension independence is satisfied, we can define the dimension service failure

probability as the proportion of outcome "failure" in a dimension $P_f^k = \sum_{U_\alpha^k(o)="failure"} T_\alpha(\beta,\delta)(o)$. The service failure probability can be represented by $P_f = 1 - \prod (1 - P_f^k)$. This implies the assumption that if anyone of the dimensions fails, then the total service fails. The service with the lowest failure probability can be chosen. Note that one can consider this model as a special case of expected utility model in which the utility mapping has only two values, "success" and "failure".

Let us assume that entity α adopts the utility mapping functions of Table 6 and the situational trust of Table 5. The "quality of food" dimension failure probability can be calculated as follows:

P_f^1= sum over all $T_\alpha(\beta,\delta)(o)$ in dimension "quality of food" whose $U^{(1)}(o)<0$

= 0

Similarly, the "service" failure probability P_f^2 has the value 0.3, and the "environment" dimension failure probability P_f^3 has the value 0.1. Therefore the total service failure probability for entity α of this restaurant service β is

$P_f=1-(1- P_f^1) (1- P_f^2) (1- P_f^3)=1-(1-0)(1-0.3)(1-0.1)=0.37$

Following the same process, entity α can calculate the utility of other restaurant services. Entity α would choose the restaurant service with the lowest total failure probability.

5 Dealing with recommendations

5.1 Discussion

While typically, the trust that a subject has is built up by his/her direct experiences, the subject may also rely on the experiences of another subject. The latter subject is called a *recommender*, and the information provided by the recommender to the former subject is called a *trust recommendation*. The recommendation may be based on direct experiences of the recommender, and also on the trust established by the recommender, possibly through recommendations obtained from other third parties.

Trust is based on a number of factors. Aside from the entity's direct experiences, another important factor is the recommendations provided by other entities of society. For this section a formal analysis of the dependency of trust on recommendation will be the central focus. It is quite common in a society that an entity could get many independent recommendations from different entities. Some of these recommendations probably conflict with each other. To address the conflict, we propose a trust combination algorithm. A recommendation need not necessarily represent the real belief of the recommending entity. Therefore recommenders may lie or give out contradictory recommendations to different entities.

How to find trustworthy recommenders is another issue. Yu and Singh [9] proposed an algorithm to find acyclic paths between a querying entity and recommenders. The number of possible paths is related to the connections between entities. If the entities are densely connected, the number of paths is quite large. If the entities are sparsely connected, the number of paths could be quite small or even zero.

Yu and Singh [9] distinguish between two kinds of beliefs: *local belief* and *total belief*. An entity's *local belief* about a correspondent is from direct experience with it and can be propagated to others upon request. An entity's *total belief (reputation)* about a correspondent combines the *local belief* (if any) with testimonies received from any witnesses. Total belief can be used for deciding whether the correspondent is trustworthy.

It is not necessary to ask for recommendations when the entity cumulates enough direct experience with certain entity in a given situation. However, it is very hard to determine whether it is enough or not. Here we propose that the number of direct experience larger than *ConfidentCount* is the right indicator of self confidence in statistic perspective. In other words, when entity α has *ConfidentCount* direct experiences with entity β in situation δ, entity α does not need to ask for recommendations for entity β in situation δ any more. However, entity α could ask for recommendations for entity β in situation δ for the purpose of verifying his/her recommenders.

5.2 Recommendation model

In the terminology of statistics, unfair recommendations refer to abnormal, biased data. In the terminology of sociology or psychology, these unfair recommendations refer to malicious or subjective recommendations. In an online community where legitimate subjective differences may exist among entities, it will be very difficult or impossible to distinguish recommendation dispersion due to subjective differences from that which is due to malicious recommendations. In this thesis, we say a recommendation is unfair when the *hyperparameters* (including *sufficient statistics* and prior belief) in the recommendation differ from those for the observation from the point view of the recommender (also called malicious recommendations), and/or when the trust distribution of the recommendation is far from that of the requesting entity[21] from the point view of the requesting entity (also called subjective difference). The requesting entity does not know exactly whether a single recommendation is fair or not, and the cause of each unfair recommendation. The requesting entity uses statistical analysis and machine learning algorithms to detect and avoid these likely unfair recommendations.

Before we study the recommendation models[22], we need to recall the definition of the utility mapping function U in the previous sections. The purpose of the utility mapping function is to map a multi-dimensional vector (trust, recommendation) to a scalar (utility), which makes it possible to sort according to a goodness relationship. Note that U_i is a subjective function which belongs to the recommender i. We denote by R_i the recommendation in the form of $R_i=\{\alpha_1'+n_1',\ldots,\alpha_r'+n_r'\}$ where $r\geq2$, $\{\alpha_1',\ldots,\alpha_r'\}$ represent the recommender's prior belief, and $\{n_1',\ldots,n_r'\}$ represent the recommender's past experience. We note that in this recommendation there is an imagined trust $E(\boldsymbol{\theta}_i')$, the expectation of $\boldsymbol{\theta}_i'$ with respect to the distribution $p(\boldsymbol{\theta}_i'|D_i', \xi_i)$. D_i' is the imagined data set inferred from recommendation R_i. We note that the trust $E(\boldsymbol{\theta}_i)$ (the expectation of $\boldsymbol{\theta}_i$ with respect to the distribution $p(\boldsymbol{\theta}_i|D_i, \xi_i)$) is derived from the prior belief $\{\alpha_1,\ldots, \alpha_r\}$ and the

[21] The requesting entity is also the trusting entity.

[22] In this thesis, we study the recommendation models from the point view of the recommenders since it is easy and unambiguous for classification.

past experience $\{n_1,\dots,n_r\}$. In our trust model, we highlight the following three recommendation models:

1. Normal recommendation

We say a recommendation is normal if $\{\alpha_1'+n_1',\dots,\alpha_r'+n_r'\}=\{\alpha_1+n_1,\dots,\alpha_r+n_r\}$. Obviously, we have $U_i(E(\boldsymbol{\theta}_i')) = U_i(E(\boldsymbol{\theta}_i))$.

2. Unfairly high recommendation

We say a recommendation is unfairly high if $U_i(E(\boldsymbol{\theta}_i')) > U_i(E(\boldsymbol{\theta}_i))$. Dellarocas [12] calls "ballot stuffing".

3. Unfairly low recommendation

We say a recommendation is unfairly low if $U_i(E(\boldsymbol{\theta}_i')) < U_i(E(\boldsymbol{\theta}_i))$. Dellarocas [12] calls "bad-mouthing".

One may think that the recommendation classification is too sensitive to the relationship of the two utility values. In this case, one may introduce a small tolerance threshold ε. If $|U_i(E(\boldsymbol{\theta}_i')) - U_i(E(\boldsymbol{\theta}_i))| < \varepsilon$, then the recommendation is thought to be a normal recommendation.

Whitby *et al*. [4] point out that “the risk of unfair recommendations is highest when they can be used to manipulate the trust system to an entity's advantage. For instance, a buyer may collude with a seller to badmouth the seller's competitors, resulting in gains to the seller [4]. By careful construction of the trust system, this risk can be diminished, by either eliminating the incentive to lie, or increasing the cost of doing so. An example of the former technique is to hide the identity of entities from each other [12]. In a sufficiently large market, this eliminates the possibility of badmouthing another entity. An example of the latter technique is to allow buyers to rate sellers only after a transaction is conducted, and charge a small amount for every transaction, thus raising the cost of ballot-stuffing or bad-mouthing. This approach is taken by eBay [4].”

In many systems, however, these techniques cannot be used. In these cases, it is desirable to be able to automatically detect and avoid unfair recommendations.

5.3 Unfair recommendation detection and avoidance

We propose that the requesting entity uses the Weighted Majority Algorithm to detect and avoid the likely unfair recommendations. The basic idea is to discount each recommendation according to its relative weight, which will increase if the recommendation is fair and will decrease if the recommendation is unfair. Suppose there are f recommenders from which to query recommendations. Each recommender gives a recommendation $R_i=\{\alpha_{i,1}+n_{i,1},\dots, \alpha_{i,r}+n_{i,r}\}$ $(i=1,\dots,f)$ which are hyperparameters of the *posterior* distribution $p(\boldsymbol{\theta}_i|D_i,\xi_i)=Dir(\boldsymbol{\theta}_i|\alpha_{i,1}+n_{i,1},\dots, \alpha_{i,r}+n_{i,r})$ based on the recommender's past experience. Note that it is impossible for the requesting entity to distinguish the subjective hyperparameters $\alpha_{i,1},\dots,\alpha_{i,r}$ from the *sufficient statistics* $N_i=\{n_{i,1},\dots, n_{i,r}\}$. Since these recommenders are not 100% trustworthy, their recommendations are discounted according to their relative weights. We refer to these discounted recommendations as *equivalent samples*, which are treated as if they are the requesting entity's own experience. For the requesting entity, the *sufficient statistics* for the *equivalent sample* of recommendation R_i are $N_i'= R_i\times w_i/s$ where $s=\sum^{f}_{i=1}w_i$ and w_i is the weight of recommender i. The final *sufficient statistics* are $N_m=N+\sum N_i'$ and the final *posterior* distribution of the requesting entity is:

$$p(\boldsymbol{\theta}_m|D_m, \xi)=Dir(\boldsymbol{\theta}_m|\alpha_1+n_1+ \sum(\alpha_{i,1}+n_{i,1})\times w_i/s,\dots, \alpha_r+n_r+ \sum (\alpha_{i,r}+n_{i,r})\times w_i/s). \qquad \textbf{(32)}$$

In the terminology of WMA, we say that $E(\boldsymbol{\theta}_i)$ is the recommender's algorithm (the expectation of $\boldsymbol{\theta}_i$ with respect to the distribution $p(\boldsymbol{\theta}_i|D_i, \xi_i)$), and $E(\boldsymbol{\theta}_m)$ is the *master algorithm* (the expectation of $\boldsymbol{\theta}_m$ with respect to the distribution $p(\boldsymbol{\theta}_m|D_m, \xi)$).

In the update step of WMA, the requesting entity estimates the correct label $\boldsymbol{\rho}$ using $E(\boldsymbol{\theta})$ (the expectation of $\boldsymbol{\theta}$ with respect to the distribution $p(\boldsymbol{\theta}|D, \xi)$). The requesting entity now can update each recommender's weight using the factor $F=1-(1-\beta)\frac{|E(\boldsymbol{\theta}_i)-E(\boldsymbol{\theta})|}{\sqrt{2}}$, note that $E(\boldsymbol{\theta}_m)$, $E(\boldsymbol{\theta}_i)$ and $E(\boldsymbol{\theta})$ are r-dimensional vectors, and $|E(\boldsymbol{\theta}_i) - E(\boldsymbol{\theta})|$ is the Euclidean distance[23]. The requesting entity uses $E(\boldsymbol{\theta})$ as the correct label $\boldsymbol{\rho}$, because there does not exist a service that can provide a correct label $\boldsymbol{\rho}$. Therefore, for the requesting entity, the only trustworthy way to get this label is to

[23] $\boldsymbol{\theta}_i$ are the parameters of the posterior distribution in recommendation R_i.

$\boldsymbol{\theta}_m$ are the parameters of the posterior distribution in master algorithm.

$\boldsymbol{\theta}$ are the parameters of the posterior distribution from past experience.

predict $\boldsymbol{\rho}$ purely based on his/her own past experience.

5.4 Example

We take a car wash example to illustrate the previous steps. To simplify our analysis, we assume that the outcome space is a 1-dimensional space, and the outcome (binomial) is either "good"(1) or "bad"(0). We can use a scalar quantity, the probability of outcome "good", *p(x=1)*, to represent trust. We assume the hyperparameters α_g, α_b of the prior distribution for every entity are 1 and 1, respectively, which implies that N_{init}=2 and the basic trust value is 1/2. This corresponds to two initial experiences with outcomes one "good" and one "bad". Car wash β is the service provider, and entity α is the service requestor.

Now entity α can calculate its initial situational trust as follows:

$$\textit{Initial situational trust } T_\alpha(\beta,\text{"car wash"})(\text{"good"}) = p(x=1) = 1/2 \qquad \textbf{(33)}$$

$$\textit{Initial situational trust } T_\alpha(\beta,\text{"car wash"})(\text{"bad"}) = p(x=0) = 1/2$$

Suppose entity α has seven experiences (n=7) denoted by an observation data set $\boldsymbol{D}=\{X_1=x_1,\ldots,X_7=x_7\}$. The *sufficient statistics* for the data set $\boldsymbol{D}$ are $\boldsymbol{N}=\{2,5\}$, which means that two "good" outcomes and five "bad" outcomes have been actually observed. Now entity α can estimate its situational trust as follows:

$$T_\alpha(\beta,\text{"car wash"})(\text{"good"}) = p(1) = (2+1)/(7+2)=1/3 \qquad \textbf{(34)}$$

$$T_\alpha(\beta,\text{"car wash"})(\text{"bad"}) = p(0)=(5+1)/(7+2)=2/3$$

Suppose entity α has two friends with weights w_1=0.2 and w_2=0.8. Friends give recommendations $\boldsymbol{R}_1=\{6,2\}$ (p(1)=0.75) and $\boldsymbol{R}_2=\{3,7\}$ (p(1)=0.3) respectively. Entity α can get the *equivalent samples* as follows:

$$\boldsymbol{N}_1' = \{6,2\}*0.2/(0.2+0.8) = \{1.2, 0.4\} \qquad \textbf{(35)}$$

$$\boldsymbol{N}_2' = \{3,7\}*0.8/(0.2+0.8) = \{2.4,5.6\}$$

With the *equivalent samples*, entity α can calculate the final *posterior* distribution as follows:

$$p(\boldsymbol{\theta}_m|\boldsymbol{D}_m, \xi)=Dir(\boldsymbol{\theta}_m|1+2+1.2+2.4,1+5+0.4+5.6)=Dir(\boldsymbol{\theta}_m|6.6,12) \qquad \textbf{(36)}$$

The situational trust can be estimated as follows:

$$T_\alpha(\beta,\text{"car wash"})(\text{"good"}) = p(1) =6.6/(6.6+12)=0.355 \qquad \textbf{(37)}$$

$T_\alpha(\beta, \text{"car wash"})(\text{"bad"}) = p(0) = 12/(6.6+12) = 0.645$

Suppose after the transaction, the actual outcome is "bad", then entity α can update its *sufficient statistics* $\boldsymbol{N}=\{2,6\}$. Entity α estimates the correct label $\rho=2/(2+6)=0.25$. The updating factor F for friend 1 is $F=1-0.5*|0.75-0.25|=0.75$. The new weight w_1 is equal to $0.2*0.75 = 0.15$. Similarly entity α can get the updating factor F for friend 2: $F=1-0.5*|0.3-0.25|=0.975$. The new weight w_2 is $0.8*0.975=0.78$.

5.5 Decision making

In our previous Section 2.4, we introduced the Expected Utility Theory (EUT). In Section 4.5, we introduced several utility models for rational decision making. All these utility models can be used here. The basic idea in a utility model is to map the multi-dimensional trust distribution to a utility value, which can be sorted according to a goodness relationship. The service provider with the highest ranking according to the goodness relationship will be chosen. If several service providers are in the highest ranking, then the deciding entity randomly selects one among them. We note that the utility function is subjective (i.e. is personal to the deciding entity).

6 Simulation

We describe several simulation experiments to illustrate the effectiveness and robustness of our trust model and to learn to what extent the weighting technique of the WMA is efficient to detect unfair recommendations. We present the following simulations: (1) Single service provider simulation, (2) Multiple service providers simulation, and (3) Bootstrapping process simulation.

6.1 Simulation description and configuration

To simplify our analysis, we assume that the outcome space is a 1-dimensional space, and the outcome (binomial) is either "good"(1) or "bad"(0). We can use a scalar quantity, the probability of outcome "good", $p(x=1)$ or p as a shorthand, to represent trust. We assume the utility mapping function $U(x) = x$ for all entities. Therefore the decision criterion is to choose the service provider with the highest trust value, i.e. the highest probability of a "good" outcome.

We assume the hyperparameters α_g, α_b of the *prior* distribution for every entity are 1 and 1, respectively, which implies that N_{init}=2, the *prior* distribution is *Beta*(1,1), and the basic trust value is 1/2. This corresponds to two initial experiences with outcomes one "good" and one "bad". The simulation includes three car washes, which are the service providers, and 250 car owners, which are the service requestors. The performance of these car washes is fixed in the sense that the probability of outcome "good" has a fixed value for each car wash. The goal of the car owners is to choose the best car wash in their opinion and to estimate the performance of car washes with few mistakes. We define a transaction as a process that includes the following actions: car owner decides that his/her car needs a wash, queries recommendations, waits for recommendations, makes decision to select the best car wash, and has the car washed by the selected car wash. After each transaction, the car owner updates his/her experience and weights according to the actual outcome. The simulation is based on a Java discrete event simulation package - *javaSimulation* [21].

Other parameters are defined as follows:

1. Each car owner uses exactly 6 randomly selected recommenders.
2. For each car owner, we set all initial weights $w_i = 1$ and $\beta = 0.5$ for the WMA.
3. Performance of car wash 1, 2, and 3 are 0.6, 0.2, and 0.4, respectively.

In the simulation, we use the following recommendation models:

1. Normal recommendation: recommendation $R=\{\alpha_g+n_g, \alpha_b+n_b\}$ where $\{n_g, n_b\}$ are the *sufficient statistics* for the observation data set
2. Unfair high recommendation: recommendation $R=\{\alpha_g+n_g+n_b, \alpha_b\}$ in which all outcomes are "good"
3. Unfair low recommendation: recommendation $R=\{\alpha_g,\alpha_b+n_g+n_b\}$ in which all outcomes are "bad"

Briefly, the simulations proceed as two phases: the bootstrap phase and the steady-state phase. The bootstrap phase consists of 200 car owners and 5,000 transactions (refer to Section 6.4 for more details). After these 200 cars have conducted 5000 transactions in total, the simulation shifts to the steady-state phase. In this phase, 50 more new car owners enter the system without any prior experience. Each of these 50 car owners conducts 250 transactions for a total of 12,500 transactions. Meanwhile, the other 200 car owners will conduct approximately 50,000 transactions, which are not recorded. During each transaction, the car owner queries all his/her 6 recommenders and predicts the possible outcomes using his/her *master algorithm*. After the transaction, the simulation records the average trust value, the average relative weights for normal recommendations and unfair recommendations, and the average hit rate, which is defined as the probability that a given car wash is chosen. The simulation repeats 30 times with different random seeds to reduce any unexpected effects of the pseudo-random generation function.

6.2 Single service provider simulation

The first question is whether our trust model is really helpful for accurate predictions and for the detection and protection against unfair recommendations. To answer this question, we assume that only car wash 1 is available. We have defined the following scenarios:

1. Scenario 1: All car owners are honest, and give normal recommendations
2. Scenario 2: 20% of car owners give unfairly low recommendations
3. Scenario 3: 40% of car owners give unfairly low recommendations[24]
4. Scenario 4: 20% of car owners give unfairly high recommendations
5. Scenario 5: 40% of car owners give unfairly high recommendations[25]

The following figures show the trust value and relative weights for the 50 new car owners. The number on the x-axis is the transaction number for each of the car owners. The number on the y-axis represents the trust value, or relative weights.

[24] There certainly exist scenarios which have higher percentage of unfairly low recommendations, such as 90% or 100%. We will discuss these scenarios in the Comparison of Section 6.5 .

[25] There certainly exist scenarios which have higher percentage of unfairly high recommendations, such as 90% or 100%.

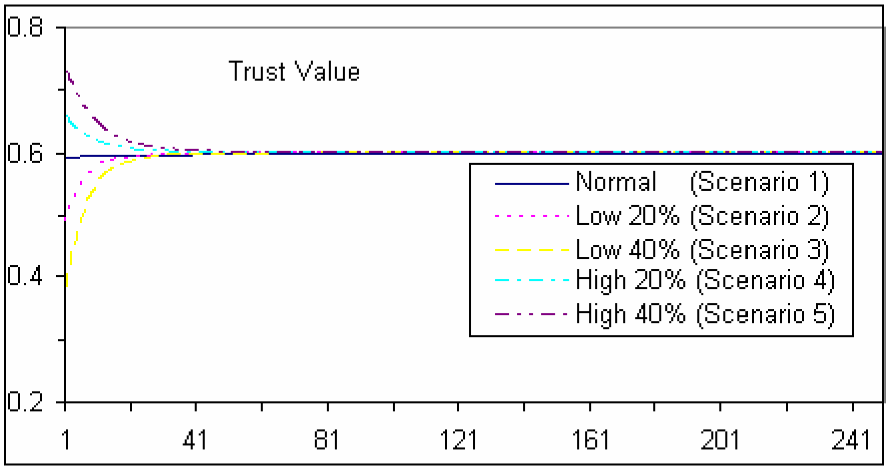

Figure 7: Trust value in Scenarios 1 through 5 (single)

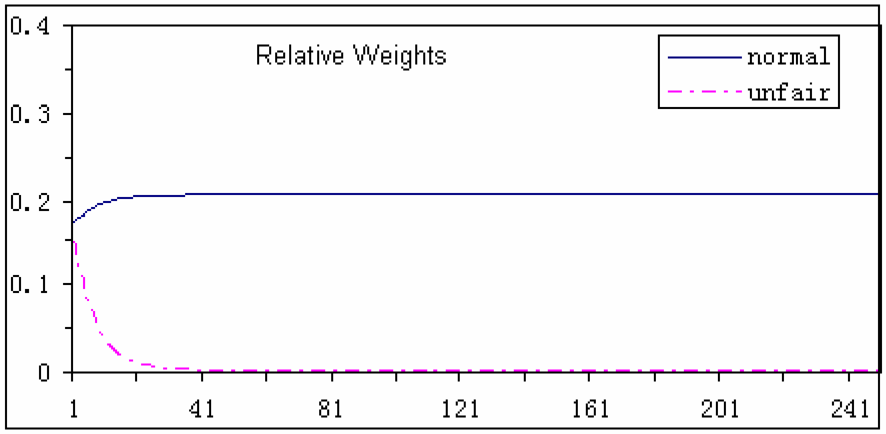

Figure 8: Relative weights in Scenario 2 (single)

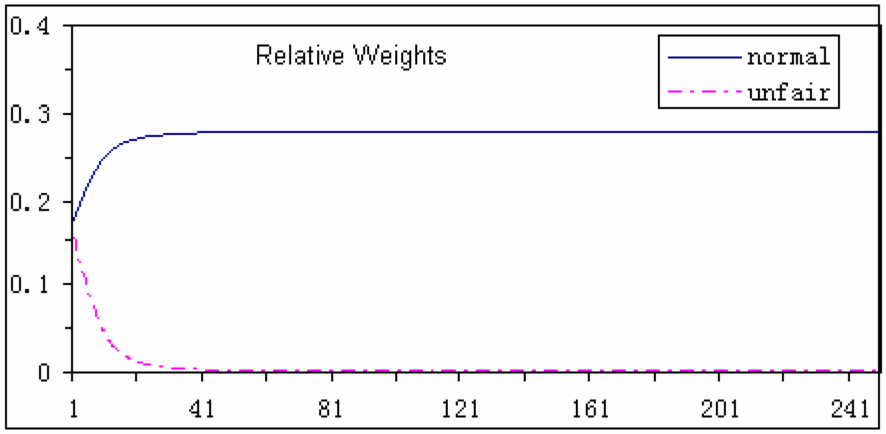

Figure 9: Relative weights in Scenario 3 (single)

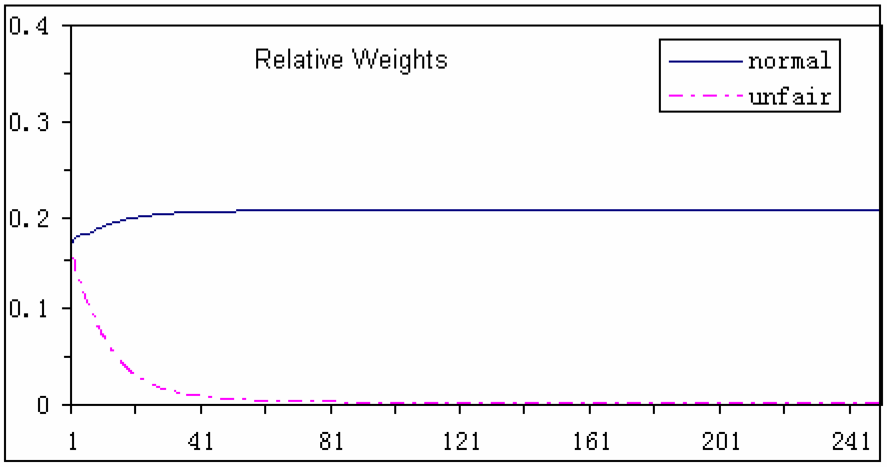

Figure 10: Relative weights in Scenario 4 (single)

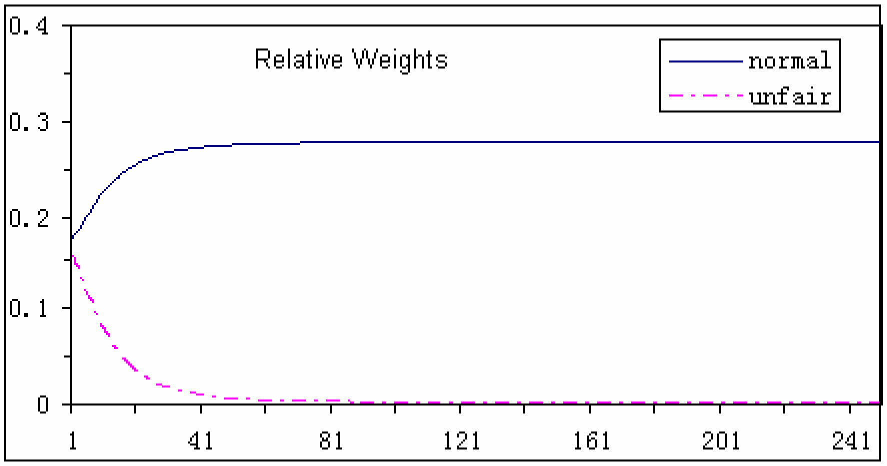

Figure 11: Relative weights in Scenario 5 (single)

Figure 7 shows that the requesting entity can make accurate predictions through WMA in all scenarios 1 through 5. We find that the average estimation error (estimated value minus correct value) becomes less then 0.05 after 10 transactions in scenarios 1 to 5.

These simulations illustrate that our trust model can effectively detect and avoid the unfair recommendations. In scenarios 2 to 5, the difference between the relative weights of fair and unfair recommendations is significant. The relative weights of unfair recommendations constantly decrease to 0. We find that the unfair recommendations do not affect the prediction accuracy much after 10 transactions. The WMA does filter out the unfair recommendations by decreasing the relative weights of unfair recommendations. We note that the parameter β for the WMA is a factor in determining the rate at which unfair recommendations are effectively removed from the car owner's decision-making process in scenarios 2 to 5.

The relative weights of fair recommendations can be estimated as follows: We take Scenario 2 as an example. For each requesting entity, there are on average 6*80% = 4.8 recommenders who give fair and normal recommendations and 6*20%=1.2 recommenders who constantly give unfair recommendations. After sufficient number of transactions, the absolute weight of each fair recommendation is approximately equal, say w. The absolute weight of each unfair recommendation is approximately 0. With these assumptions, we can estimate the relative weights of fair recommendations as follows:

$$\text{Relative weights} = w/(4.8*w + 1.2*0) = 1/4.8 \approx 0.21.$$

The result is confirmed by Figure 8. In fact, in this simple scenario, we even can estimate the percentage of recommenders who give normal recommendations from the relative weights figure. From Figure 8, we see that the relative weight of fair recommendations is approximately 0.2 and of unfair recommendations is approximately 0. Suppose the percentage of recommenders who give normal recommendations is q. We use the same formula to calculate the relative weight

$$\text{Relative weight} = w/(6*q*w + 6*(1-q)*0)=0.2$$

We can solve the above equation for q and obtain q=83%, which is pretty close to the real percentage 80%. The same methods can be used for other scenarios.

6.3 Multiple service providers simulation

The second question is whether our trust model is really helpful for decision-making. We now assume that all three car washes are available. The simulation configuration is the same as in the previous section except that there are now three car washes. We have defined the following scenarios:

1. Scenario 1: All car owners are honest, and give normal recommendations
2. Scenario 2: 20% of car owners give unfairly low recommendations to car wash 1 and normal recommendations to the other car washes
3. Scenario 3: 40% of car owners give unfairly low recommendations to car wash 1 and

normal recommendations to the other car washes

4. Scenario 4: 20% of car owners give unfairly high recommendations to car wash 2 and normal recommendations to the other car washes
5. Scenario 5: 40% of car owners give unfairly high recommendations to car wash 2 and normal recommendations to the other car washes

The following figures show the trust value, hit rate, and relative weights for the 50 new car owners. The number on the x-axis is the transaction number for each of the car owners. The number on the y-axis represents the trust value, hit rate, or relative weights.

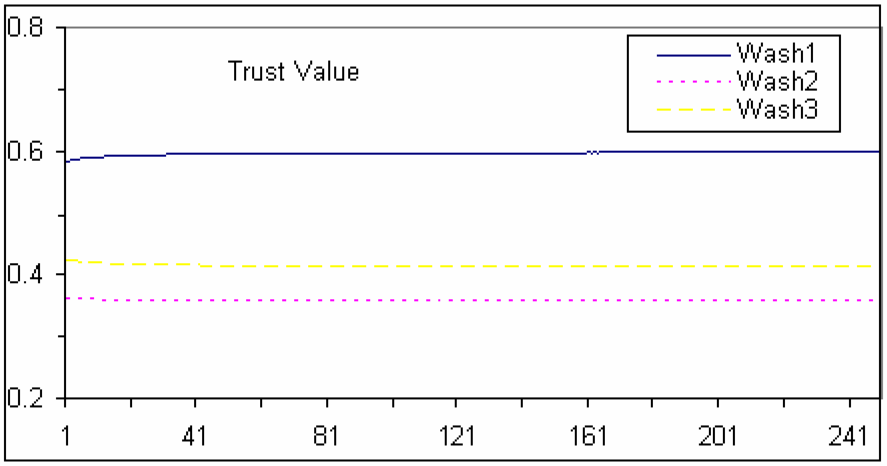

Figure 12: Trust value in Scenario 1 (multiple)

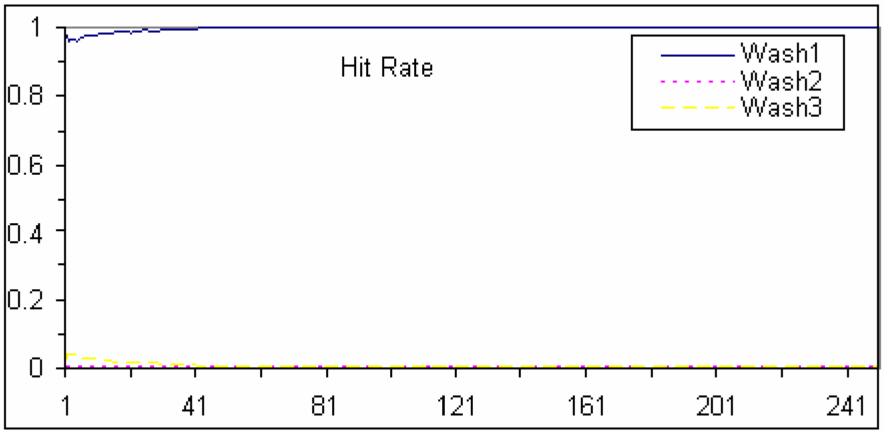

Figure 13: Hit rate in Scenario 1 (multiple)

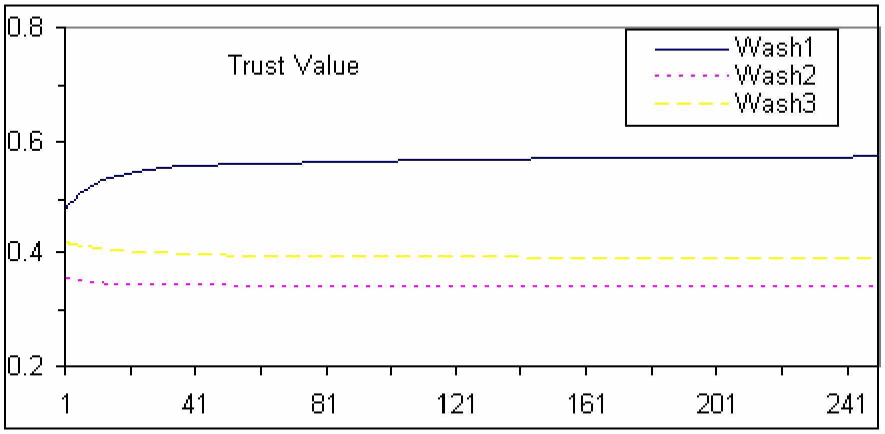

Figure 14: Trust value in Scenario 2(multiple)

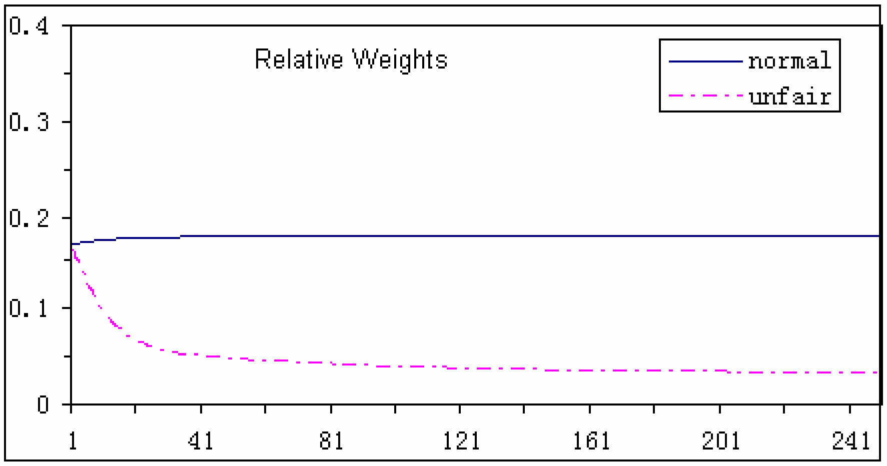

Figure 15: Relative weights in Scenario 2 (multiple)

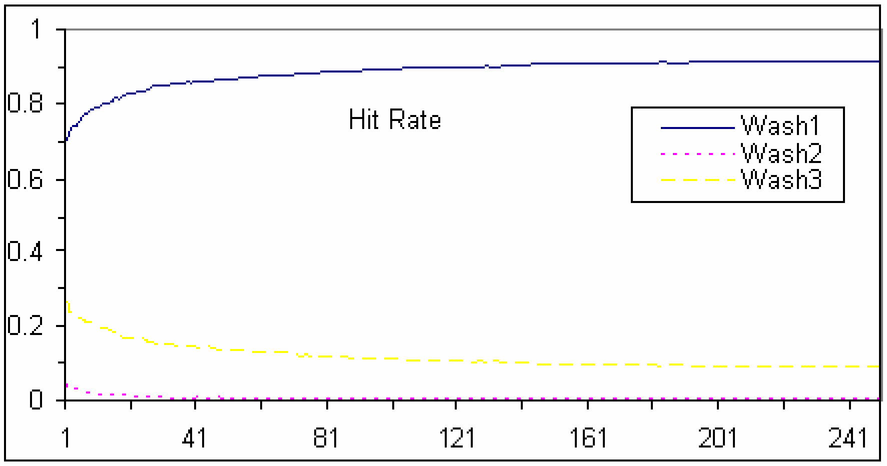

Figure 16: Hit rate in Scenario 2 (multiple)

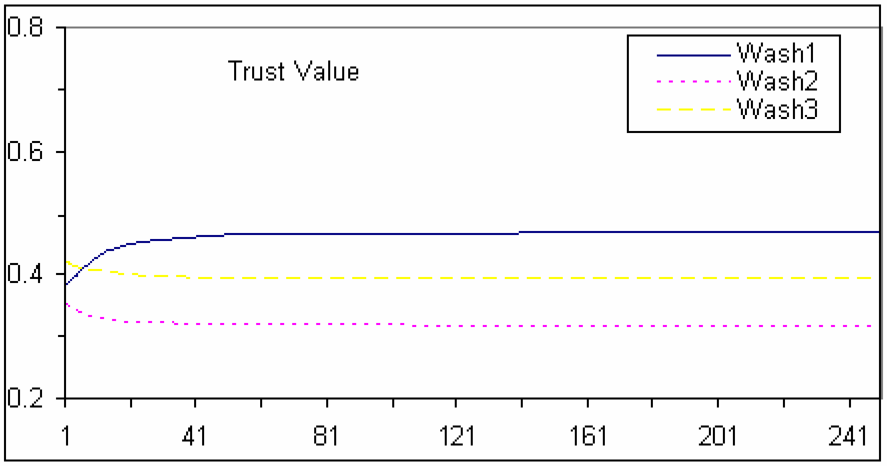

Figure 17: Trust value in Scenario 3 (multiple)

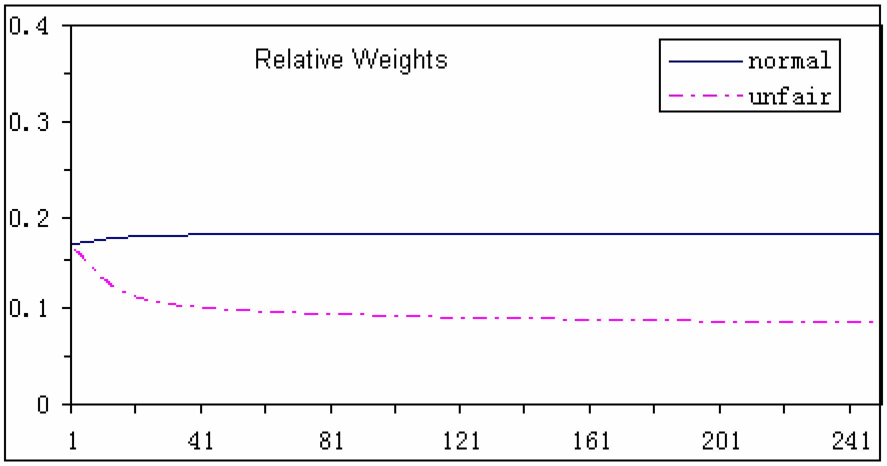

Figure 18: Relative weights in Scenario 3 (multiple)

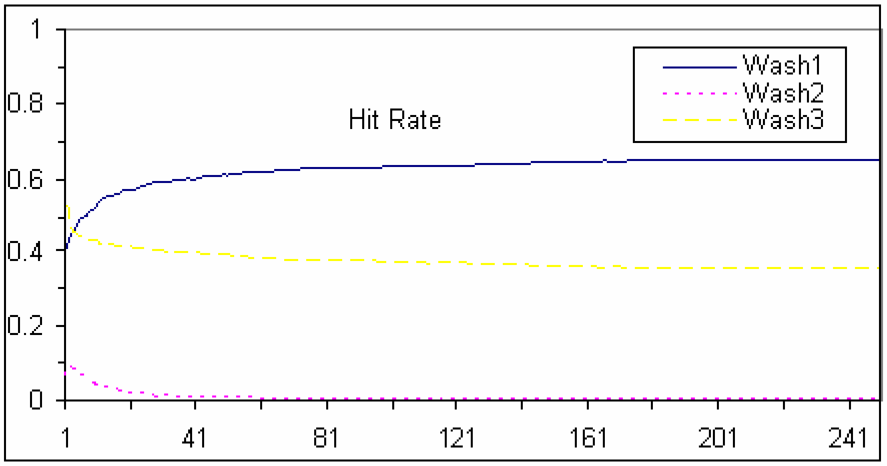

Figure 19: Hit rate in Scenario 3 (multiple)

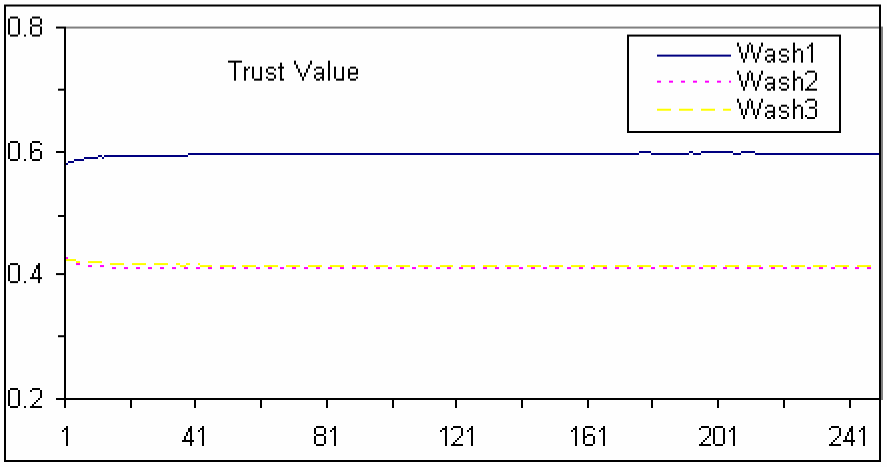

Figure 20: Trust value in Scenario 4 (multiple)

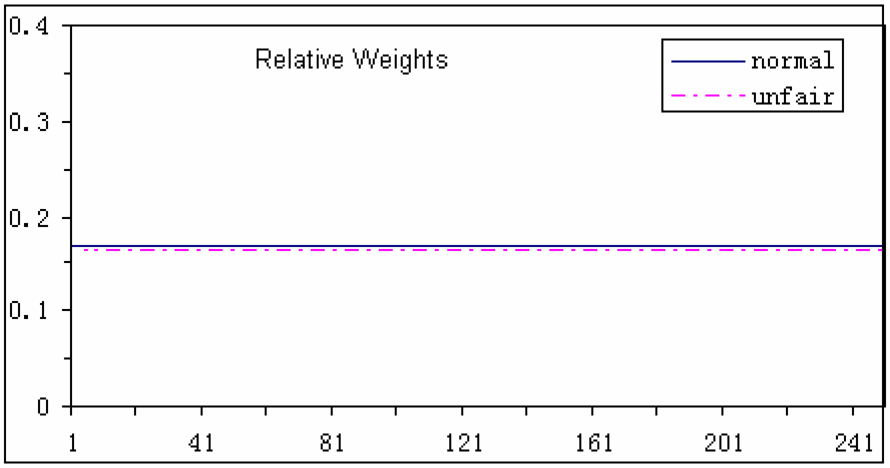

Figure 21: Relative weights in Scenario 4 (multiple)

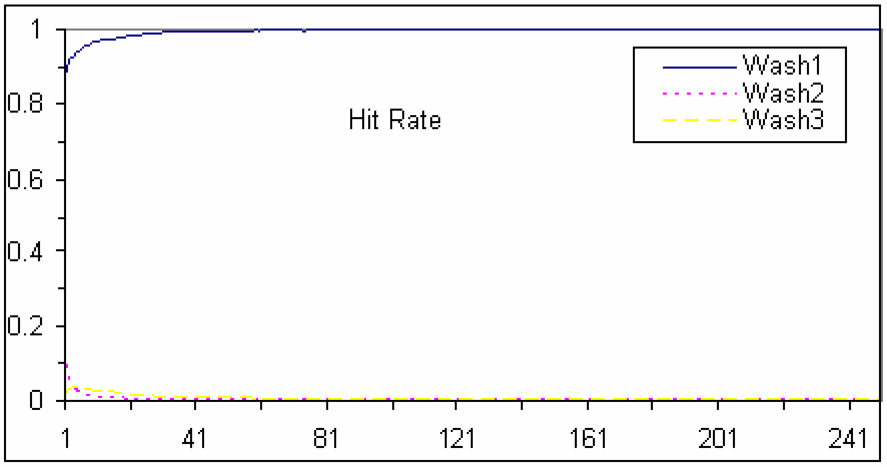

Figure 22: Hit rate in Scenario 4 (multiple)

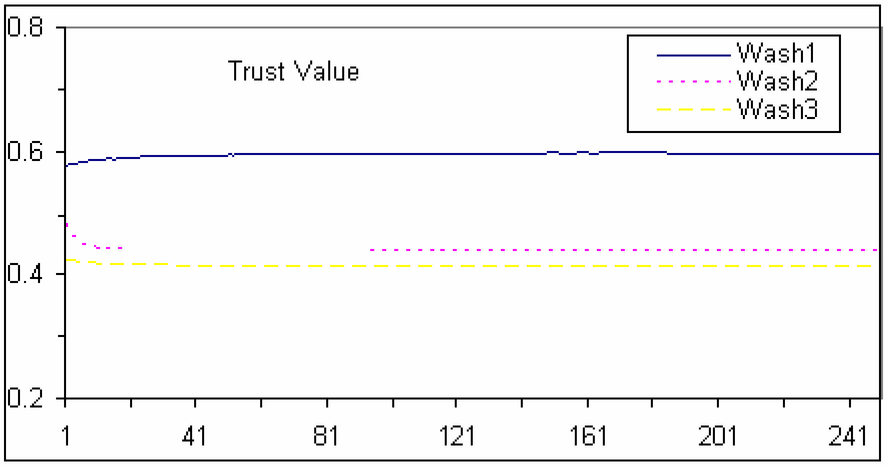

Figure 23: Trust value in Scenario 5 (multiple)

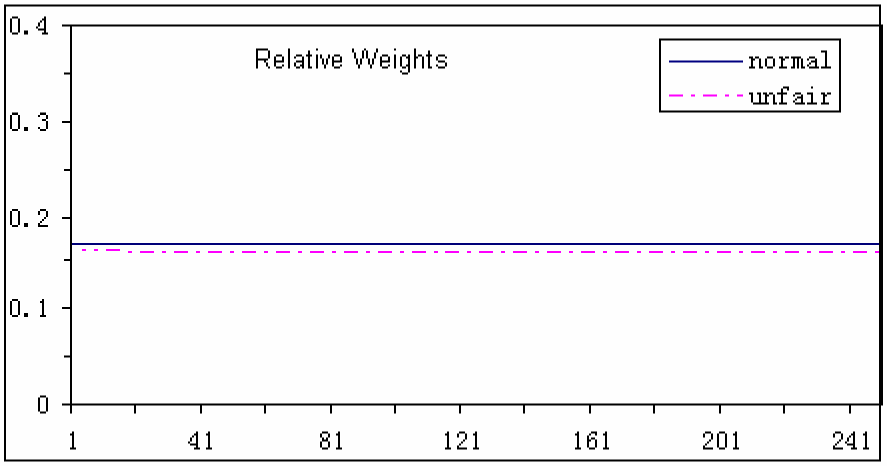

Figure 24: Relative weights in Scenario 5 (multiple)

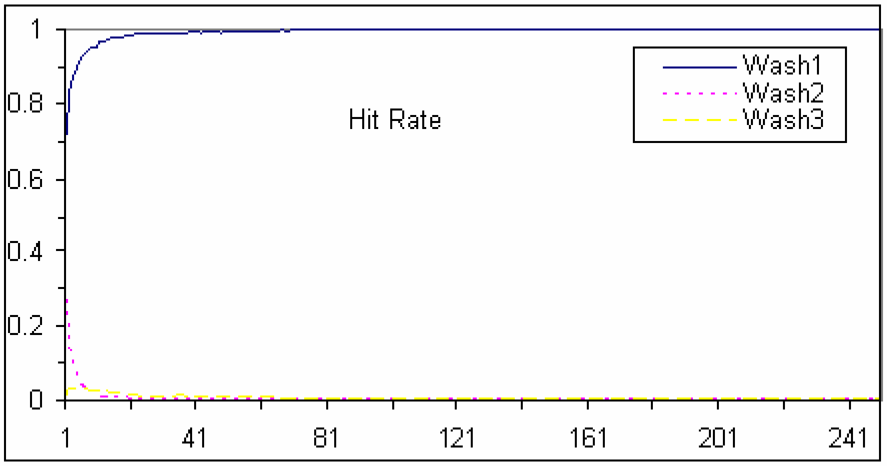

Figure 25: Hit rate in Scenario 5 (multiple)

Figure 12 shows that the car owners can make accurate prediction in Scenario 1 (multiple), especially for car wash 1 and 3. However the trust value of car wash 2 has a positive bias 0.16. This is because the car owners have very few numbers of transactions with car wash 2. Thus the prior beliefs (basic trust value is higher than the performance of car wash 2) have an important impact on the trust value estimation. Therefore the trust value of car wash 2 is over estimated.

Figure 13, Figure 16, Figure 22 and Figure 25 tell us that the average hit rate of car wash 1 in scenarios (multiple) 1, 2, 4, and 5 is very close to 100%, which is what we expect. Figure 19 tells us that the average hit rate of car wash 1 is close to 70% in Scenario 3 (multiple). This bad performance is due to a combination of two reasons. First, the 40% unfair low recommendations significantly lower the trust value for car wash 1, especially when the requesting entity only has 5 or fewer transactions (actually the average trust value for car wash 1 is less than that of car wash 3 in the first 5 transactions). Second, the low trust value for car wash 1 consequently lowers the chance of being chosen. The requesting entity is blinded by these unfair low recommendations. Therefore the requesting entity has a limited chance to select car wash 1 and cannot discern correctly the quality of the recommendations for car wash 1. Figure 18 confirms that the relative weights of unfair recommendations are quite high compared to those in scenarios 2, 4, and 5 (multiple).

Figure 21 shows that the average relative weights for normal and unfair recommendations are almost identical in Scenario 4 (multiple). This is because 20% unfair high recommendations for car wash 2 only make its trust value rise to 0.35, which is not high enough compared with car wash 1 and 3. Therefore the requesting entity only has very limited chance to select car wash 1, which makes the requesting entity have very limited chance to update the relative weights regarding car wash 1. The same reason also explains Figure 24.

6.4 Bootstrapping process simulation

The simulation in this section shows the bootstrapping process of our trust model. How fast can our trust model reach its steady state? In other words, the average trust of service provider should depend on its performance.

This simulation consists of three car washes and 50 car owners. These 50 car owners enter the system without any prior experience. Each of these 50 car owners conducts 250 transactions for a total of 12,500 transactions. During each transaction, every car owner queries all his/her 6 recommenders and predicts the possible outcomes using his/her *master algorithm*. After the transaction, the simulation records the average trust value, the average relative weights for normal recommendations and unfair recommendations, and the average hit rate. The rest of the simulation configuration is the same as for the previous simulations.

The scenarios are the same as in Section 6.3. The following figures show the trust value, hit rate, and relative weights for the 50 car owners. The number on the x-axis is the transaction number for each of the car owners. The number on the y-axis represents the trust value, hit rate, or relative weights.

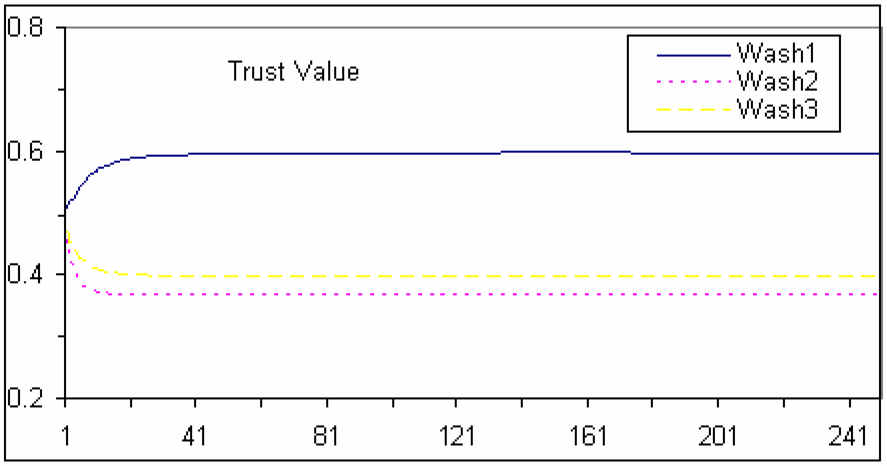

Figure 26: Trust value in Scenario 1 (bootstrap)

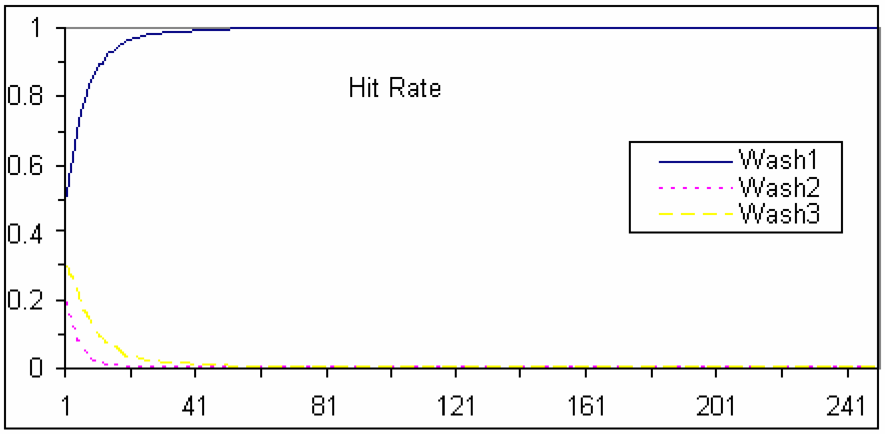

Figure 27: Hit rate in Scenario 1 (bootstrap)

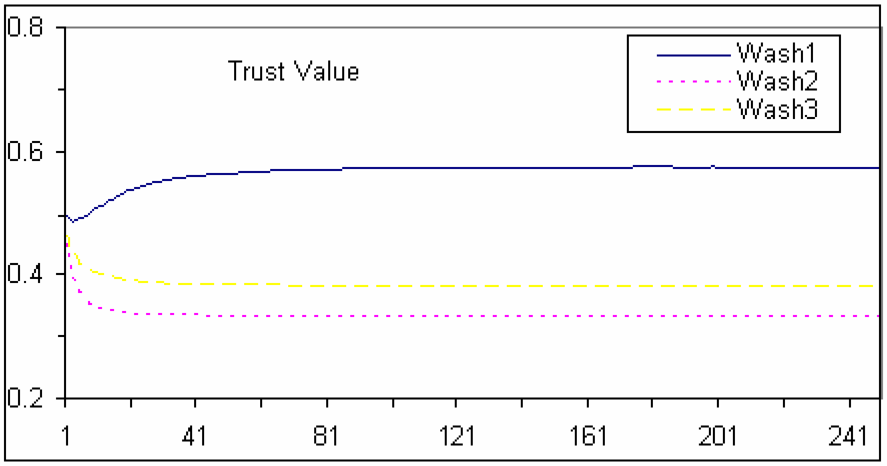

Figure 28: Trust value in Scenario 2 (bootstrap)

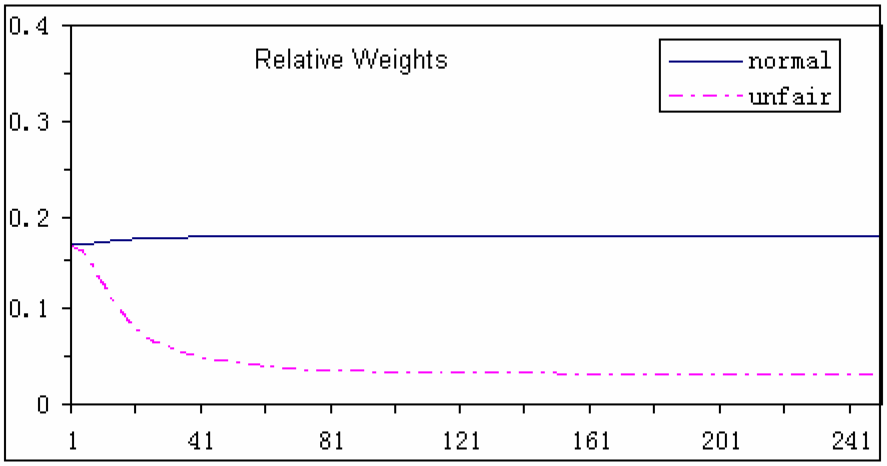

Figure 29: Relative weights in Scenario 2 (bootstrap)

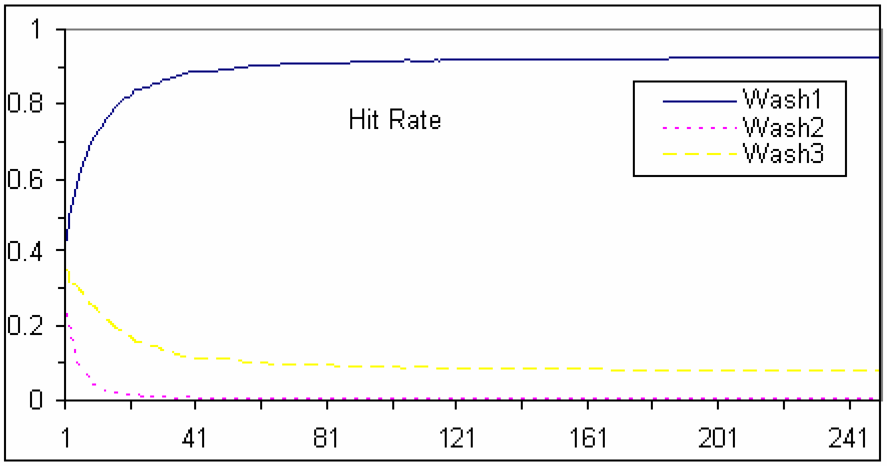

Figure 30: Hit rate in Scenario 2 (bootstrap)

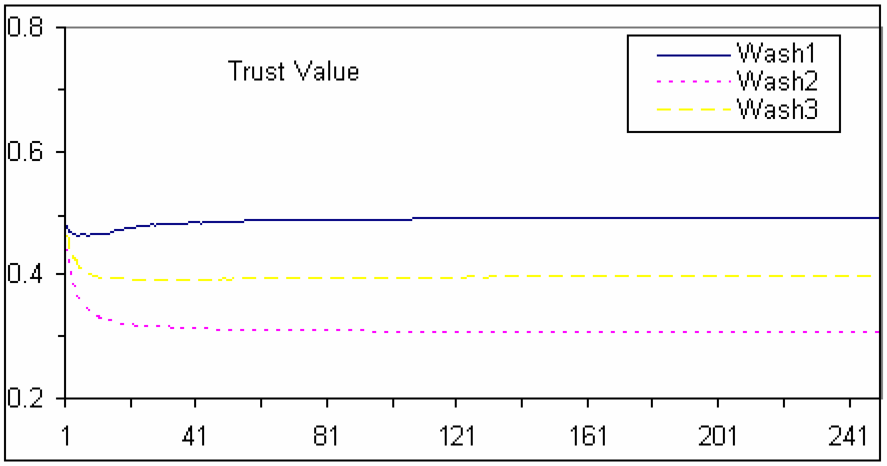

Figure 31: Trust value in Scenario 3 (bootstrap)

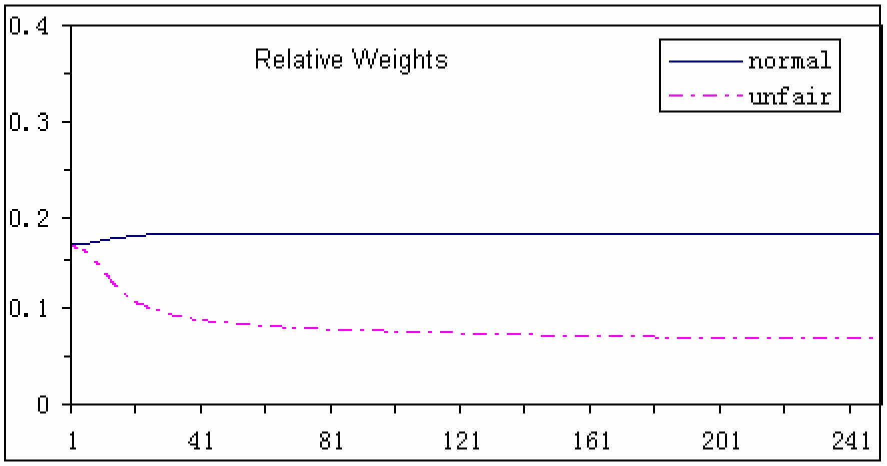

Figure 32: Relative weights in Scenario 3 (bootstrap)

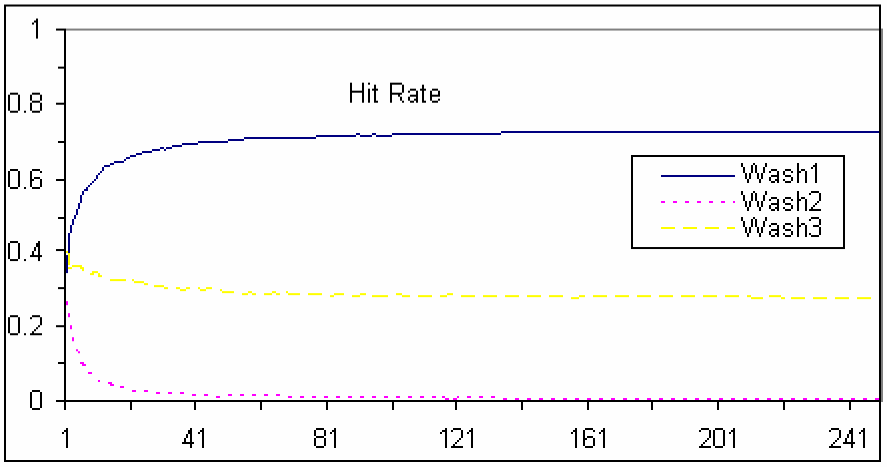

Figure 33: Hit rate in Scenario 3 (bootstrap)

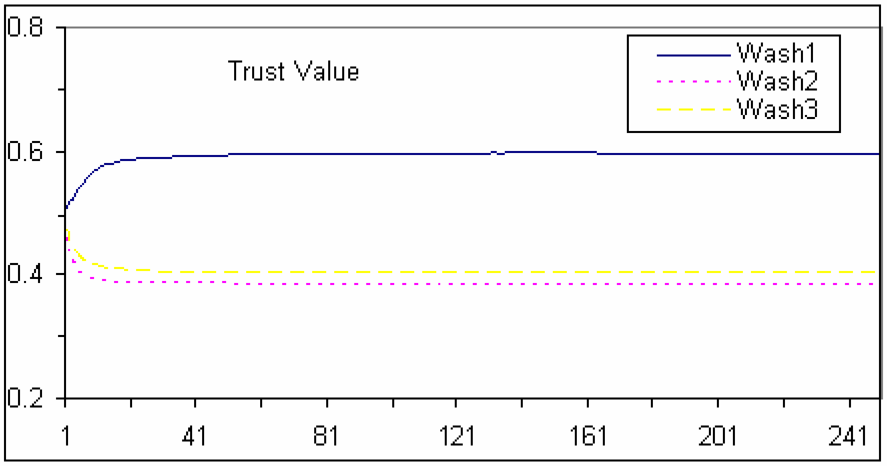

Figure 34: Trust value in Scenario 4 (bootstrap)

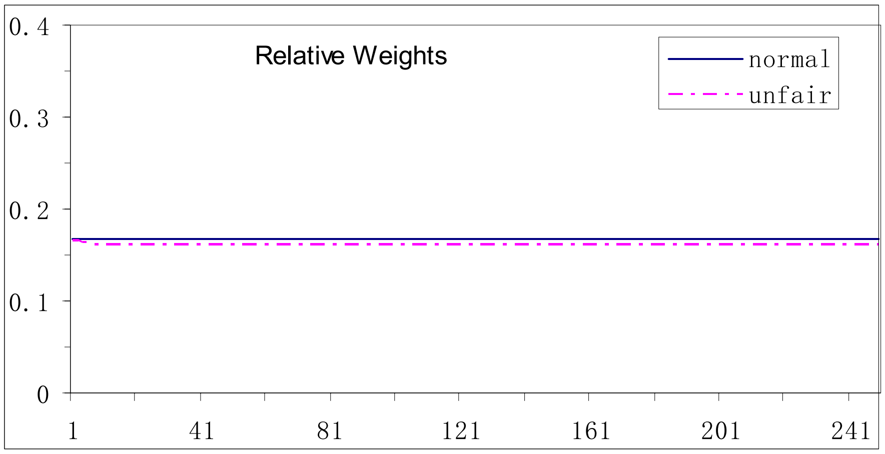

Figure 35: Relative weights in Scenario 4 (bootstrap)

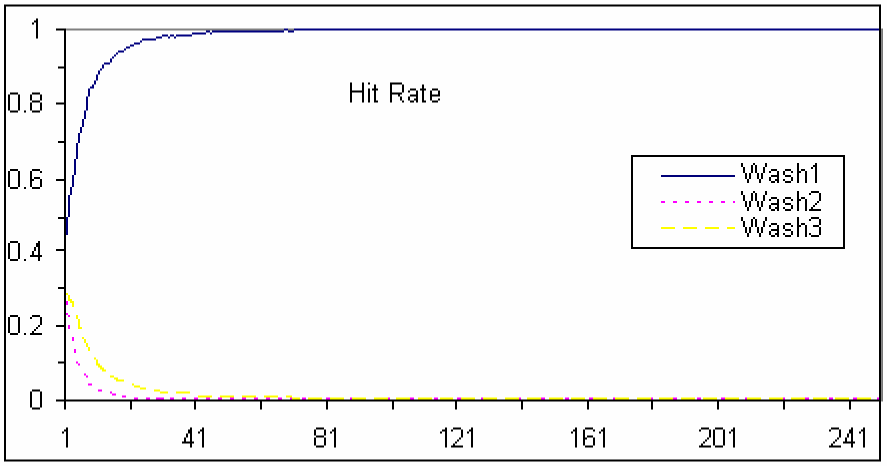

Figure 36: Hit rate in Scenario 4 (bootstrap)

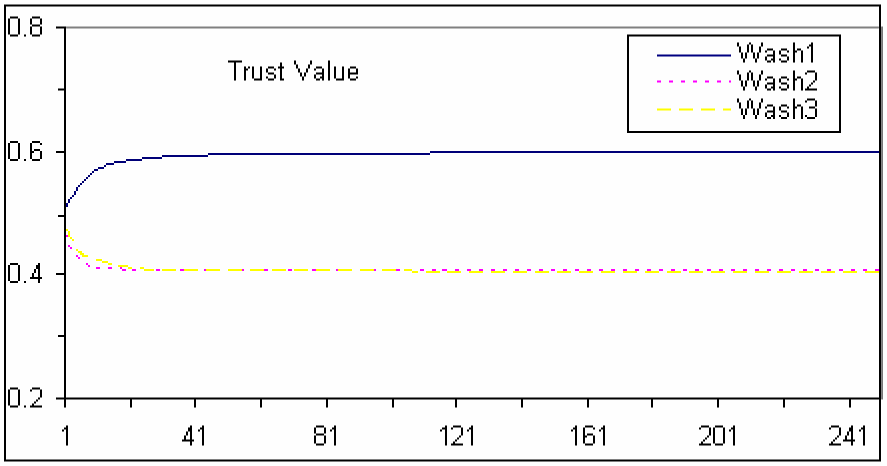

Figure 37: Trust value in Scenario 5 (bootstrap)

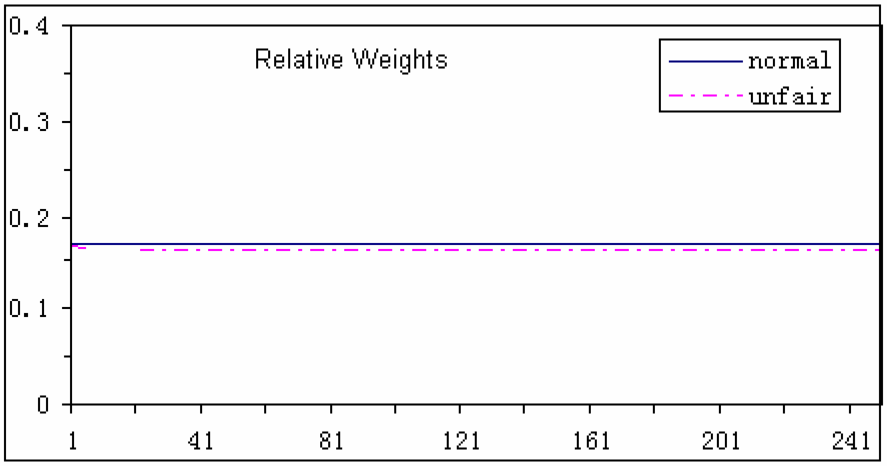

Figure 38: Relative weights in Scenario 5 (bootstrap)

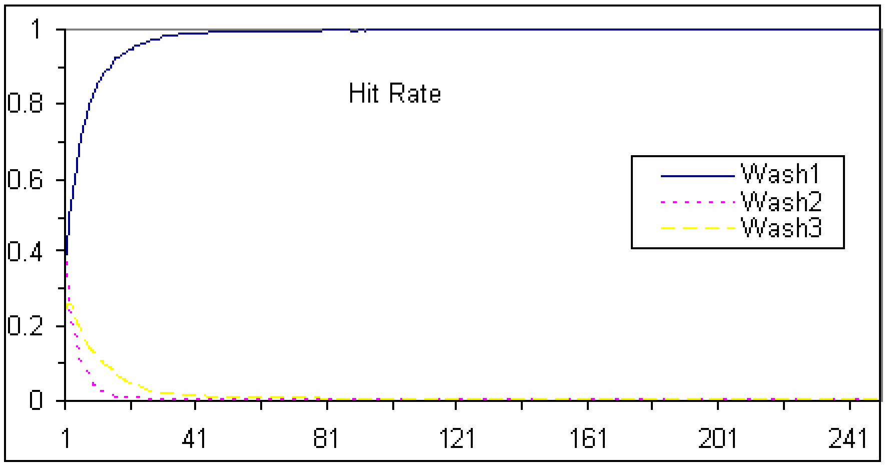

Figure 39: Hit rate in Scenario 5 (bootstrap)

The simulation results are similar to their counterparts in the Section on Multiple service providers simulation and the explanations are the same as those in that section. Compared with the Multiple service providers simulation, the difference is that there are no extra experienced car owners (in the previous simulations, there were 200 of these). Since all the car owners enter the system without prior experience, the basic trust has an important impact on their trust and recommendation in the first several transactions. This contributes to some difference in the figures. The simulations show that the average trust, relative weights, and hit rate stay nearly constant after 40 transactions (and, in fact, change very slowly after only 20 transactions).

6.5 Comparison

We continue the previous car wash simulations. The purpose of the simulation is to illustrate the accuracy of the proposed recommendation system when the performance of the service provider may vary over time. Our simulations also provide a comparison of our recommendation system with the Iterated Filtering Algorithm (IFA) proposed by Whitby *et al.* [4]. The simulation includes one car wash, which is the service provider, and 50 car owners, which are the service requestors. The simulation is divided into sessions, and each session is divided into transactions. After each session, the service provider adapts its performance *Perf*, which is the probability of outcome "good".

6.5.1 Forgetting factor

Since the performance of the car wash changes over time, the basic assumption about a given outcome distribution for the actions of the entity, valid over all time, is not true any more. In this case, we must give different weights to the different experiences. It is desirable to give greater weight to more recent experiences. This can be achieved by introducing a forgetting factor γ where $0 \le \gamma \le 1$. When a new experience yielding outcome x^k is observed, the sufficient statistics $\boldsymbol{N}$ will be updated as follows:

$$\begin{aligned} \boldsymbol{N} &:= \boldsymbol{N} \times \gamma \\ n_k &:= n_k + 1 \end{aligned} \tag{38}$$

In our simulation, we set the forgetting factor γ=0.7.

6.5.2 Service provider behavior

Once the car wash has committed to a transaction, it will either provide a “good” service (with a probability equal to its performance) or a “bad” service (with a probability equal to 1 – performance). At the end of each session t, the performance for the next session (t+1) is chosen randomly as follows:

$$Perf_{t+1} = \begin{cases} Perf_t + delta & with\ a\ probability\ equal\ to\ 1/3 \\ Perf_t - delta & with\ a\ probability\ equal\ to\ 1/3 \\ Perf_t & with\ a\ probability\ equal\ to\ 1/3 \end{cases} \tag{39}$$

In the simulation, $delta$ = 0.1. In addition, the performance $Perf_{t+1}$ is restricted to the range [0,1]. We note that this kind of service provider behavior is a Markovian random walk.

6.5.3 Service requestor behavior

Car owners are divided into three groups: (1) In the fair group, the car owners always give fair recommendations. (2) In the unfairly high group, each car owner selects one favorite car wash, and gives unfairly high recommendations with a probability equal to *Punfair*. (3) In the unfairly low group, each car owner selects one target car wash, and gives unfairly low recommendations with a probability equal to *Punfair*. In this thesis, we only represent one car wash scenario. We note that there is no decision making process involved in one car wash scenario, while for multiple car wash scenarios, the decision making process is crucial.

6.5.4 Recommendation model

We define three recommendation models in the simulation: (1) Fair recommendation, for each trust value x, the recommendation given is $y=x$. (2) Unfairly high recommendation, for each trust value x, the recommendation is $y=x+a*(1-x)$. (3) Unfairly low recommendation, for each trust value x, the recommendation is $y=(1-a)*x$. Here the const a is an exaggeration coefficient ($0 \leq a \leq 1$), which represents the degree of unfairness. Variable x is the estimation of the performance of the car wash. In this thesis, we only represent simulation results of fair and unfairly low recommendations.

6.5.5 Simulation results

The simulations are conducted to assess the effectiveness of our proposal in several scenarios. For the purpose of comparison, all scenarios are conducted in three different modes: WMA, IFA, and SIMPLE. In the WMA mode, the entities use our proposal to combine recommendations. In the IFA mode, the entities use IFA (Iterated Filtering Algorithm by Whitby *et al*. [4]) to filter out unfair recommendations. In the SIMPLE mode, which is the reference mode, the entities simply average all the recommendations.

All simulations are conducted over 2000 transactions (20 sessions of 100 transactions each). The simulation is based on a Java simulation package - *javaSimulation* [21]. Several initial parameters are defined as follows:

1. Each car owner has exactly 6 randomly selected recommenders.
2. For each car owner, we set the initial weights w_i=1 and β=0.5 for WMA, and *quantile*=0.01 for IFA.
3. Initial performance of car wash 1 is 0.6.

The following figures show the average trust value and the average estimation error for the car wash in the first 2000 transactions. The number on the x-axis is the transaction number of the car wash. The number on the y-axis represents average trust value, performance, and average estimation error, which is defined as average trust value minus performance.

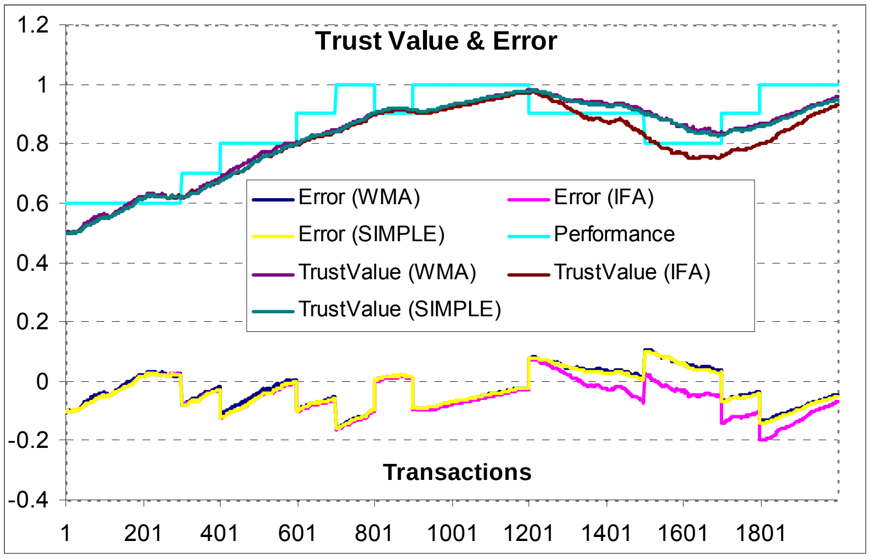

Figure 40: Trust value, performance and error with 0% unfair recommenders

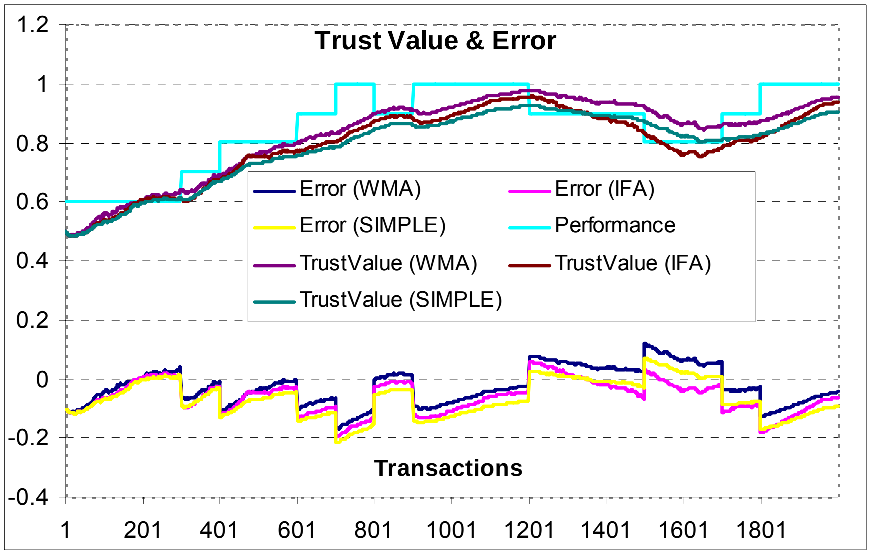

Figure 41: Trust value, performance and error with 20% unfair recommenders

Figure 40 is the basic scenario, in which there are no unfair recommenders. Figure 41 consists of 20% unfairly low recommenders with *Punfair*=100% and exaggeration coefficient *a*=0.875. Figure 42 consists of 40% unfairly low recommenders with *Punfair*=100% and exaggeration coefficient *a* =0.875.

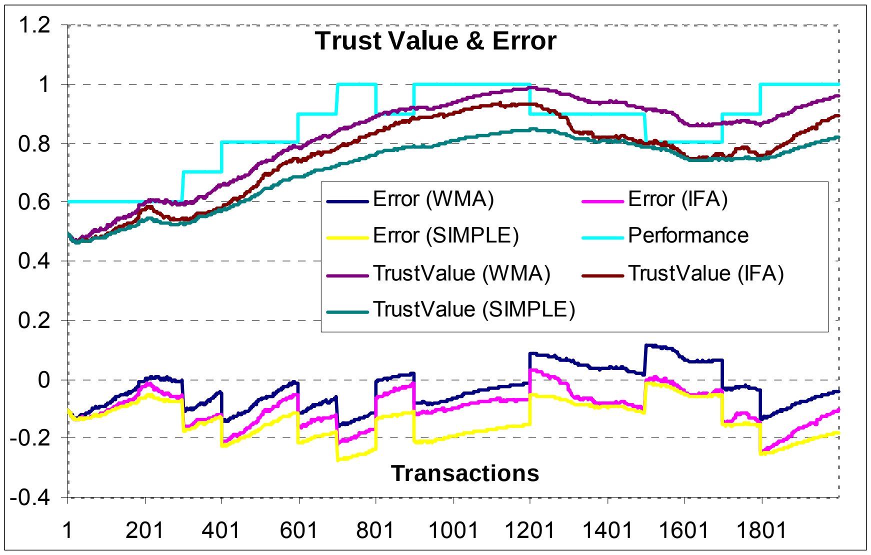

Figure 42: Trust value, performance and error with 40% unfair recommenders

Figure 40 shows that in the ideal scenario, the WMA and IFA [4] algorithms are not helpful, since there are no unfair recommendations. Actually in transactions 1700-2000, the average estimation error in the IFA mode is worse than that in the SIMPLE mode, which implies that the IFA mistakenly filters out fair recommendations. Figure 41 and Figure 42 show that both the WMA and the IFA can detect and avoid unfair recommendations to some degree. These figures directly illustrate how the trust value follows the changing performance in different modes.

6.5.6 Algorithm effectiveness

The effectiveness of each algorithm is examined by simulating scenarios with three parameters:

1. The proportion of unfair recommenders
2. The probability that unfair recommenders give unfair recommendation (*Punfair*)
3. The degree of unfairness of unfair recommendations (exaggeration coefficient *a*)

For ease of comparison, we summarize the mean and standard deviation of average estimation error of WMA, IFA, and SIMPLE algorithms in different scenarios.

Unfairly low rate		0	0.2	0.4	0.6	0.8	1.0
mean	WMA	-0.0267	-0.0264	-0.0323	-0.0497	-0.1123	-0.3943
	IFA	-0.0546	-0.0610	-0.0987	-0.2624	-0.5854	-0.7203
	SIMPLE	-0.0310	-0.0661	-0.1412	-0.2394	-0.4475	-0.6415
std dev	WMA	0.0607	0.0629	0.0686	0.0734	0.0789	0.0772
	IFA	0.0558	0.0550	0.0592	0.0674	0.1509	0.1567
	SIMPLE	0.0611	0.0629	0.0639	0.0715	0.0950	0.1270

Table 11: Statistical information about average error (*α*=0.875, *Punfair*=1)

The following figure shows the mean and standard deviation of average estimation error of WMA, IFA, and SIMPLE algorithms during the first 2000 transactions in different scenarios described in Table 11. The number on the x-axis is the unfairly low rate (proportion of unfair recommenders). The number on the y-axis represents the mean and standard deviation of average estimation error. We note that the negative mean value of the average estimation error is due to the fact that the performance increases for most of the sessions.

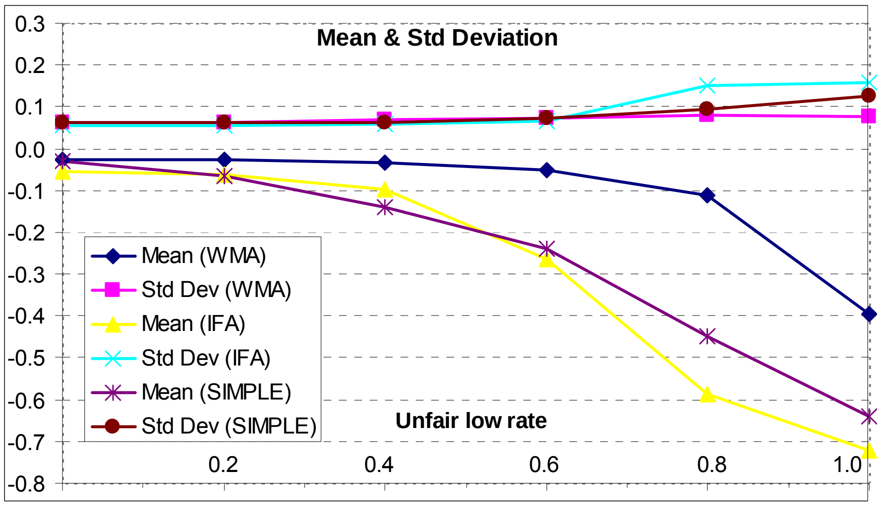

Figure 43: Mean and standard deviation of average estimation error

It is clear in Figure 43 that the standard deviation of average estimation error of the three algorithms is not significantly different. However the standard deviation of the IFA algorithm increases faster than the others when the unfairly low rate is greater than 60%. Our proposal (WMA) has the least mean average estimation error (abstract value).

Table 12 and Figure 44 show the effectiveness of the three algorithms in scenarios with exaggeration coefficient *a*= 0.875 and *Punfair*=0.25. Not surprisingly, all the algorithms increase their effectiveness compared with those in Table 11 and Figure 43. It is shown that IFA algorithm is the worst when the unfairly low rate is less than 40%. WMA is the most effective algorithm in these scenarios.

Unfairly low rate		0	0.2	0.4	0.6	0.8	1.0
mean	WMA	-0.0267	-0.0256	-0.0302	-0.0430	-0.0630	-0.1187
	IFA	-0.0546	-0.0531	-0.0542	-0.0660	-0.1117	-0.1782
	SIMPLE	-0.0310	-0.0335	-0.0550	-0.0847	-0.1285	-0.1866
std dev	WMA	0.0607	0.0626	0.0672	0.0676	0.0668	0.0598
	IFA	0.0558	0.0562	0.0609	0.0605	0.0582	0.0601
	SIMPLE	0.0611	0.0635	0.0657	0.0655	0.0657	0.0623

Table 12: Statistical information about average error (α=0.875, *Punfair*=0.25)

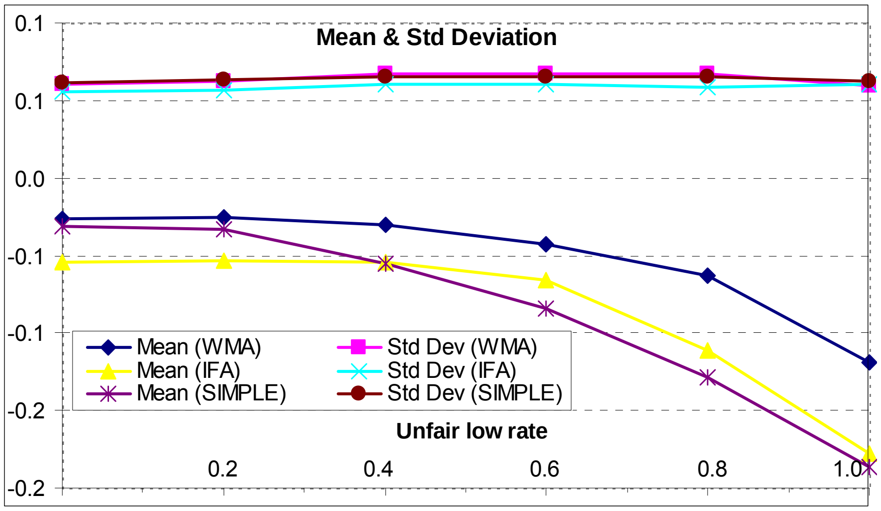

Figure 44: Mean and standard deviation of average estimation error

The following two figures show the effectiveness in scenarios with exaggeration coefficient *a*= 0.2, *Punfair*=0.25, and exaggeration coefficient *a*= 0.99, *Punfair*=1, respectively.

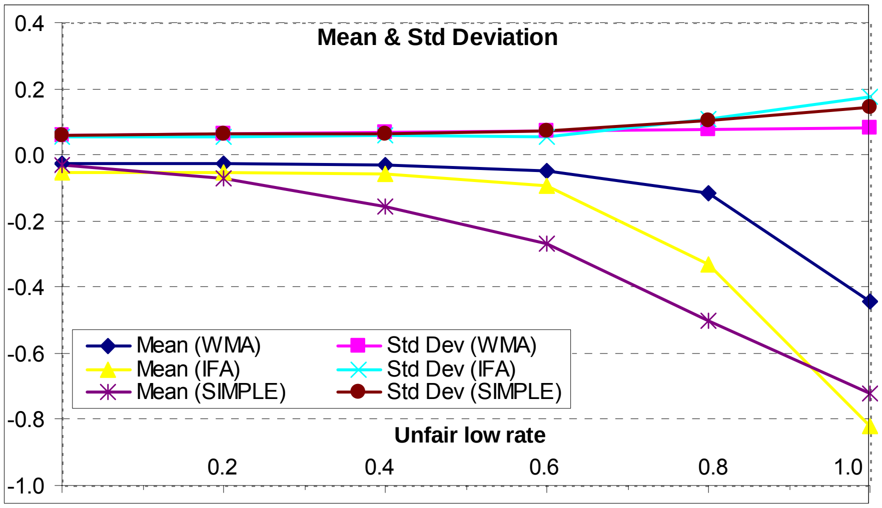

Figure 45: Mean and standard deviation of average error (α=0.2, *Punfair*=0.25)

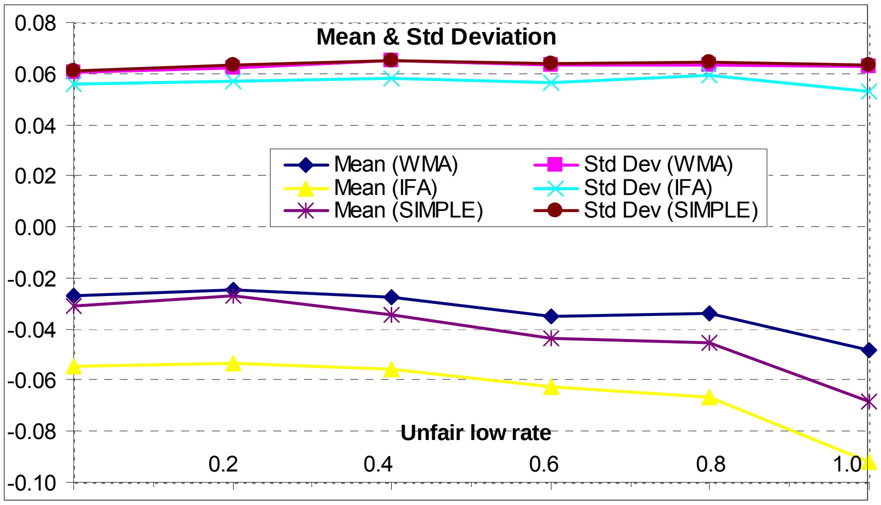

Figure 46: Mean and standard deviation of average error (α=0.99, *Punfair*=1)

Table 11 and Figure 43 show that the IFA [4] algorithm is not effective compared with the SIMPLE algorithm in the scenario with exaggeration coefficient *a*=0.875. In the scenario with low exaggeration coefficient and low *Punfair*, the IFA is particularly ineffective, even worse than the SIMPLE algorithm, which is illustrated by Figure 45. Figure 46 shows that the IFA algorithm is most effective in the scenario with very high exaggeration coefficient *a*= 0.99. To summarize, the IFA algorithm is very sensitive to the exaggeration coefficient. The higher the exaggeration coefficient, the more effective the IFA algorithm is. Regarding the average estimation error (absolute value), our proposal (WMA) is better than IFA in all the above scenarios. More interestingly, our proposal is also much faster than IFA, since IFA consists of complex lower and upper *quantile* calculations. In our simulations, we also changed the parameter *quantile* of the IFA algorithm several times and found that the IFA algorithm produces the best results in most situations when the parameter *quantile* has the value 0.01.

6.6 Summary

From the simulation, we find that unfair high recommendations have no long run effect on our trust model, because the requesting entity can make accurate estimations based on his/her own experience; therefore, the requesting entity can detect and avoid unfair high recommendations. However the unfair low recommendations can have a long run effect in our trust model, if and only if the unfair low recommendations can alter the ranking of service providers as illustrated in Scenario 3 (multiple). There may be situations in which unfair recommendations cannot be automatically detected, but their effect is reduced over time.

In the Single service provider simulation of Section 6.2, the requesting entities can make accurate prediction in all the scenarios. This is because there is only one available car wash. The car owner can accumulate own experience and make accurate prediction. And there is virtually no decision making process involved in the scenarios in this section.

In the Multiple service providers simulation of Section 6.3, the requesting entities can not make accurate prediction in all the scenarios. This is because they are facing the more complex situations: three available car washes mean that the car owners have to make a decision before each transaction. The right decisions are always based on the accurate predictions which could be affected by recommendations. The car owners struggle to filter out unfair recommendations.

In the Bootstrapping process simulation of Section 6.4, the requesting entities face almost the same problems as in Section Multiple service providers simulation.

In the Comparison of Section 6.5, we compare the effectiveness of WMA, IFA [4], and SIMPLE algorithms. The simulation results confirm that our approach can improve the accuracy of trust estimation in the scenarios with changing performance.

7 Conclusions and future work

7.1 Conclusions

We have addressed the problem of building a general trust model for online entities based on their direct experiences and the recommendations of other entities. Considering that trust is a complex and multi-faceted thing, we use the estimated distribution in a multidimensional outcome space to represent trust. The statistical characterizations of trust (Bayesian estimation, Dirichlet distribution, outcome space mapping) are discussed. Our trust model can be used by different decision models (utility, failure probability, satisfaction) for rational decision making in different scenarios.

We also have studied the problem of unfair recommendations. We focus on detection and protection against unfair recommendations. For simplicity, this work assumes that the unfair recommendations are consistent in the sense that recommenders will not switch from one recommendation model to another recommendation model (e.g. give an unfair high recommendation one time and an unfair low recommendation the next time). We also assume that each entity has a fixed number of recommenders. Our approach integrates Bayesian estimation with a Weighted Majority Algorithm. We have shown through simulations that it is flexible and effective in most situations. Our proposal shows great promise as a technique for improving the accuracy of trust estimation, and hence the fairness and robustness of trust models. In particular, our proposal is computationally efficient compared with the IFA [4] algorithm.

Due to the intrinsic limitations of WMA, our trust model will become ineffective when either of the following conditions holds: (1) The requesting entity has only one recommender, which makes the relative weight always equal to 1, no matter what the recommendations are. (2) Among the requesting entity's recommenders, no one gives normal recommendations. These limitations can be solved by treating the requesting entity him/herself as a recommender. Therefore, the requesting entity at least has one recommender (him/herself) who always gives normal recommendations.

In our trust model, we studied three recommendation models. However, there may be another kind of unfair recommendation: flooded recommendation, in which the experience number is enlarged deliberately by the recommender in order to circumvent the Weighted Majority Algorithm. The flooded recommendation can be avoided by imposing an upper bound on experience number in recommendations. Any recommendation whose experience number exceeds the upper bound should be scaled to the upper bound.

Our trust model is a deterministic model[26], in the sense that the output (decision-making, trust, weights) can be predicted with 100 percent certainty when the input (recommendations, self experience) is known. This deterministic feature can sometimes make our model ineffective for unfair low recommendations, as illustrated in the simulations above. To eliminate the long run effect of unfair low recommendations, a small perturbation can be introduced into the decision-making. Some possible techniques for achieving this include random decision-making at certain rate (for example, at every tenth transaction the requesting entity chooses randomly among all possible service providers) and dynamic recommender selection (for example, at every tenth transaction the requesting entity chooses a new recommender randomly and/or deletes the recommender with the least weight). We will be exploring the implications of such non-determinism on the quality of decision making in future work.

7.2 Future work

There are many questions left to answer in our proposed trust model. Among these, we highlight the following important issues.

1. Non-Determinism

We have shown that our trust model is a deterministic model and the drawbacks and

[26] From Wikipedia [38], "a deterministic system is a conceptual model of the philosophical doctrine determinism applied to a system for understanding everything that has and will occur in the system, based on the physical outcomes of causality. In a deterministic system, every action, or cause, produces a reaction, or effect, and every reaction, in turn, become the cause of subsequent reactions. The totality of these cascading events can theoretically show exactly how the system will exist at any moment in time."

the limitations. We believe that some random perturbation could improve the performance of our trust model in real situations.

2. Recommender finding

In our trust model, we assume every entity has a group of fixed recommenders. This is too restrictive for real situations. To relax this restriction, we should study the dynamic recommender selection algorithm. Another issue is what weight should be assigned initially to the new recommenders. Since the absolute weights in WMA decrease constantly, it is obviously unfair to assign the same initial weights to new recommenders.

3. Dynamic behavior

In all the simulations, we assumed that all the unfair recommendations are consistent in the sense that recommenders will not switch from one recommendation model to another recommendation model (e.g. give an unfair high recommendation one time and an unfair low recommendation the next time). This assumption is also too restrictive for real situations. The smart entity probably does not behave in such a predictable way. In other words, we should study the dynamic behavior of smart entities to see the effectiveness and robustness of our trust model.

4. Computational complexity, memory requirements and traffic efficiency

Network traffic efficiency is the main focus. Traffic efficiency is closely related to the connections between entities. In order to find enough recommendations, the entity tends to query as many other entities as possible. This will cause a serious network traffic burden if no traffic control is applied.

5. Repeatable transactions

Our approach assumes that entities can interact with each other repeatedly. In a small-size online community, this is not a problem. In a large-size one, this assumption is generally not true. Take eBay for an example: an empirical study [30] based on data from February 1 to June 30, 1999, has shown that 89% of all seller-buyer pairs conducted just one transaction during the time period, that 98.9% conducted no more than four transactions, that it is rare for a buyer to meet the same seller, and that trust is mostly built on a single experience. In that case, however, we believe that if the small world phenomenon[27] [34] occurs, then it can make our trust applicable in such large-size online community [42]. According to Jovanovic [23], the phenomenon happens in the P2P network: Gnutella. It means that people tend to interact with other people more frequently in a small sub-community than with people outside.

7.3 Comparison with other trust models

In Chapter 2, we summarize the background and related work. We review previous work on trust models, trust representation, recommendation handling, and decision making. In section 2.3, we also compare our trust model with two others briefly. In this section, we want to make a thorough comparison with these two trust models considering the simulation results.

The trust model of Whitby *et al*. [4] is a memoryless system, which calculates the combined rating based on current recommendations only. This memoryless feature means that entities can not accumulate knowledge about the quality of recommendations in the past. As illustrated in the simulations, the average reputation error will not decrease while the trusting entity repeats transactions. With 30% of entities rating unfairly, the average reputation error is around 0.1. With 40% of entities rating unfairly, Whitby *et al*.'s trust model breaks down [4]. However, since this trust model is a memoryless system, its performance is not affected by the size of the online community, in other words, the performance of Whitby *et al*.'s trust model does not depend on whether the small world phenomenon [34] occurs or not.

Unlike Whitby *et al*.'s trust model, Both Yu and Singh's trust model [9] and our trust model are non-memoryless systems in the sense that the current trust is correlated with past recommendations, current recommendations and past experience. Entities in Yu and

[27] In Wikipedia [39], "the small world phenomenon is the theory that everyone in the world can be reached through a short chain of social acquaintances. The concept gave rise to the famous phrase six degrees of separation after a 1967 small world experiment by psychologist Stanley Milgram which found that two random US citizens were connected by an average of six acquaintances. However, after more than thirty years its status as a description of heterogeneous social networks (such as the aforementioned 'everyone in the world') still remains an open question."

Singh's trust model and ours maintain a set of weights and they update weights after each transaction. By updating weights, entities accumulate knowledge about the quality of recommendations given by different recommenders. The average reputation/trust error will decrease while entities repeat transactions, which is illustrated in all simulations in our trust model. In Figure 7 for example, the average trust error in Scenario 5 (single) is around 0.13 in first two transactions, 0.08 at the 5th transaction, and 0.05 at the 10th transaction. Similarly, the simulation in Yu and Singh's trust model shows that the rating error changes from 0.31 to about 0.17 after 5,000 simulation cycles, and becomes less than 0.05 after 10,000 cycles.

Both Yu and Singh's trust model and ours are useful where two entities can repeatedly interact with each other. In other words the small world phenomenon [34] must happen, so that weights can be updated repeatedly. In our trust model, an entity needs to update weights around 10 times to ensure the average trust error is less than 0.05. In Yu and Singh's trust model, an entity needs to update weights around 20 times to ensure the average rating error is less than 0.05. We note that these values are obtained from different simulation configurations. If the small world phenomenon [34] does not happen (that is, if we assume that trust in built on less than 5 experiences), then the average trust error in our model is around 0.13 to 0.08 and the average rating error in Yu and Singh's trust model is around 0.31 to 0.25. The difference in performance of Yu and Singh's trust model and ours is due to a combination of two reasons. First, by using Bayesian estimation, entities in our model can integrate their subjective prior knowledge and the actual experience into the estimation of trust. The prior knowledge lets entities in our model have a better starting point. Second, by using a better version of WMA, entities in our model can learn faster than those in Yu and Singh's trust model [9]. Entities in Yu and Singh's trust model [9] only estimate the correct label by using three values: 0, 0.5 and 1, while entities in our model estimate the correct label by using the distribution of past experience.

References

[1] Abdelhakim Hafid, Gregor v. Bochmann: *An Approach to Quality of Service Management in Distributed Multimedia Application: Design and an Implementation*. Multimedia Tools Appl. 9(2), 1999, pp. 167-191

[2] About: *Sufficient Statistics*. Retrieved December 10, 2004 from http://economics.about.com/od/economicsglossary/g/sufficients.htm

[3] Alfarez Abdul-Rahman, Stephen Hailes: *Supporting Trust in Virtual Communities*. In Proceedings of the 33rd Hawaii International Conferences on System Sciences - Volume 6, 2000

[4] Andrew Whitby, Audun Jøsang, Jadwiga Indulska: *Filtering Out Unfair Ratings in Bayesian Reputation Systems*. In proceedings of the Workshop on Trust in Agent Societies, at the Autonomous Agents and Multi Agent Systems Conference (AAMAS2004), New York, July 2004

[5] Antony Richards, Glynn Rogers, Mark Antoniades, Varuni Vitana: *Mapping User Level QoS from a Single Parameter*. Proc. Second IFIP/IEEE International Conference on Management of Multimedia Networks and Services '98 (MMNS'98), Versailles, Nov. 1998

[6] Audun Jøsang: *A Logic For Uncertain Probabilities*. International Journal of Uncertainty, Fuzziness and Knowledge-Based Systems, Vol. 9, No. 3 (June 2001)

[7] Benjamin Yakir: *Advanced Statistical Models B(52805)* class notes. Retrieved December 10, 2004 from http://pluto.huji.ac.il/~msby/modelsb-files/modelsb03.pdf

[8] Bin Yu, Munindar P. Singh: *An Evidential Model of Distributed Reputation*

Management. In Proceedings of the first international joint conference on Autonomous agents and multiagent systems, pp. 294 – 301, July 2002

[9] Bin Yu, Munindar P. Singh: *Detecting Deception in Reputation management.* AAMAS'03, pp. 73 – 80, July 2003

[10] Carlisle Adams, Steve Lloyd: *Understanding PKI: Concepts, Standards, and Deployment Considerations, Second Edition.* Addison-Wesley, 2002

[11] Catholijn M. Jonker, Jan Treur: *Formal Analysis of Models for the Dynamics of Trust based on Experiences.* 2nd Workshop on Deception, Fraud and Trust in Agent Societies, pp. 221 – 231, 1999

[12] Chrysanthos Dellarocas: *Immunizing Online Reputation Reporting Systems Against Unfair Ratings and Discriminatory Behavior*. In proceedings of Electronic Commerce (EC'00)/ACM, Minneapolis, Minnesota, October 2000, pp. 150-157

[13] David Heckerman: *A Tutorial on Learning With Bayesian Networks*. Technical Report, March 1995. Retrieved December 10, 2004 from http://research.microsoft.com/research/pubs/view.aspx?msr_tr_id=MSR-TR-95-06

[14] Edward A. Lee, David Tse: *Memoryless and Non-Memoryless Systems*. Retrieved December 10, 2004 from http://ptolemy.eecs.berkeley.edu/eecs20/week9/systems.html

[15] Ethan Cerami: *Web Services Essentials, First Edition.* O'Reilly, February 2002

[16] Glenn Shafer: *A Mathematical Theory of Evidence*. Princeton University Press, Princeton, NJ, 1976

[17] Greg Elofson: *Developing Trust with Intelligent Agents: An Exploratory Study*. In Proceedings of the 1st International Workshop on Trust, pp. 125 – 139, 1998

[18] Hyperdictionary: *Knowledge-Based Systems*. Retrieved December 10, 2004 from http://www.hyperdictionary.com/dictionary/knowledge-based+system

[19] Jianqiang Shi, Gregor v. Bochmann, Carlisle Adams: *A Trust Model with Statistical Foundation*. In proceedings of 2nd International workshop on Formal Aspects in Security and Trust, Toulouse, France, August 2004, pp.169-181

[20] John H. Roberts, Glen L. Urban: *Modeling multiattribute utility, risk, and belief dynamics for new consumer durable brand choice*. Management Science, v.34 n.2, p.167-185, Feb., 1988

[21] Keld Helsgaun: *Discrete Event Simulation in Java*. Technical Report, Roskilde University, Denmark. Retrieved December 10, 2004 from http://www.akira.ruc.dk/~keld/research/JAVASIMULATION/JAVASIMULATION-1.1/docs/Report.pdf

[22] Lik Mui, Mojdeh Mohtashemi: *Rational Decision Making Using Social Information*. Submission to Rationality and Society, March 12, 2002

[23] M. Jovanovic: *Modeling large-scale peer-to-peer networks and a case study of gnutella*. Master thesis, University of Cincinnati, 2001

[24] Mao Chen, Jaswinder Pal Singh: *Computing and Using Reputations for Internet Ratings*. EC'01, pp. 154 – 162, October 2001

[25] Merriam-Webster Online Dictionary: *Trust*. Retrieved December 10, 2004 from http://www.merriam-webster.com/cgi-bin/dictionary?book=Dictionary&va=trust&x=23&y=12

[26] Miquel Montaner, Beatriz López, Josep Lluís de la Rosa: *Opinion-Based Filtering Through Trust*. In Proceedings of the Sixth International Workshop on Cooperative Information Agents (Madrid, Spain, September 2002), S. Ossowski and O. Shehory,

Eds., vol. 2446 of LNAI, Springer-Verlag, pp. 164-178

[27] Nick Littlestone, Manfred K. Warnuth: *The Weighted Majority Algorithm*. Information and Computation, 108(2):212-261, 1994

[28] NIST/SEMATECH *e-Handbook of Statistical Methods*. Retrieved December 10, 2004 from http://www.itl.nist.gov/div898/handbook/apr/section1/apr181.htm

[29] P. Macnaughton-Smith, W. Williams, M. Dale, L. Mockett: *Dissimilarity analysis: A New Technique of Hierarchical Sub-division*. Nature, 202:1034-35, 1964

[30] Paul Resnick, Richard Zeckhauser: *Trust Among Strangers in Internet Transactions: Empirical Analysis of eBay's Reputation System*. In NBER workshop on empirical studies of electronic commerce, 2001

[31] Philippe Mongin: *Expected Utility Theory*. Retrieved December 10, 2004 from http://expected-utility-theory.behaviouralfinance.net/Mong.pdf

[32] R.L. Keeney, H. Raiffa: *Decision Making with Multiple Objectives: Preference and Value Tradeoffs*. John Wiley and Sons, New York, NY.

[33] Richard Hughey, Anders Krogh: *Hidden Markov models for sequence analysis: extension and analysis of the basic method*. Computer Applications in the Biosciences (CABIOS), 12(2):95-107, 1996

[34] Stanley Milgram: *The small world problem*. Psychology Today 1(1), May 1967, pp.60 – 67

[35] Stephane Pauquet: *Bayesian Estimation*. Course Notes, San Francisco State University. Retrieved December 10, 2004 from http://userwww.sfsu.edu/~efc/classes/biol710/bayes/a)-Bayesian-Estimation-web.html

[36] Stephen Paul Marsh: *Formalising Trust as a Computational Concept*. Ph.D. Thesis, University of Stirling, April 1994

[37] Thomas Tran, Robin Cohen: *Learning Algorithms for Software Agents in Uncertain and Untrusted Market Environments.* In Proceedings of the Eighteenth International Joint Conference on Artificial Intelligence (IJCAI-03), pp. 1475 – 1476, August 2003

[38] Wikipedia: *Deterministic system*. Retrieved December 10, 2004 from http://en.wikipedia.org/wiki/Deterministic_system_%28philosophy%29

[39] Wikipedia: *Small_world_phenomenon*. Retrieved December 10, 2004 from http://en.wikipedia.org/wiki/Small_world_phenomenon

[40] Wikipedia: *Web Services*. Retrieved December 10, 2004 from http://www.webopedia.com/TERM/W/Web_services.html

[41] Y.H. Tan and W. Thoen: *Towards a Generic Model of Trust for Electronic Commerce.* In Proceedings of the 12th International Bled Electronic Commerce Conference, Vol. 1, pp. 346 – 359, Bled Slovenia, 1999

[42] Yao Wang, Julita Vassileva: *Bayesian Network in Peer-to-Peer Networks*. Proc. of IEEE International Conference on Web Intelligence (WI 2003), October 13-17, 2003, Halifax, Canada

Appendices

A Measures of distance

A.1 Euclidean distance

The Euclidean distance is defined as:

$\text{distance} = \sqrt{\sum_{i=1}^{r} (x_i - y_i)^2}$

r : the number of dimension

x_i : ith dimension in vector x

y_i : ith dimension in vector y

Support these x_i, y_i in *r*-dimensional vectors *x*, *y* are probabilities. We want to calculate the range of the Euclidean distance between these two vectors. This problem can be presented in mathematics as follows:

$\textit{objective} : \max \sqrt{\sum (x_i - y_i)^2}$

constraints :

$\sum x_i = 1 \quad (i = 1...r)$

$\sum y_i = 1 \quad (i = 1...r)$

$x_i \geq 0, y_i \geq 0$

Proof:

$$\sqrt{\sum (x_i - y_i)^2} = \sqrt{\sum (x_i^2 + y_i^2 - 2x_i y_i)} \leq \sqrt{\sum (x_i^2 + y_i^2)} \leq \sqrt{\sum (x_i + y_i)} = \sqrt{2}$$

Therefore, the range of the Euclidean distance between these two vectors is $[0, \sqrt{2}]$.

A.2 Minkowski metrics

The Minkowski metric is defined as:

$$\text{distance} = \sqrt[t]{\sum_{i=1}^{r} | x_i - y_i |^t}$$

r : the number of dimension

x_i : ith dimension in vector x

y_i : ith dimension in vector y

t = 1 Hamming distance, t = 2 Euclidean distance, t = 3 cubic distance

Support these x_i, y_i in *r*-dimensional vectors *x*, *y* are probabilities. We want to calculate the range of the Minkowski distance between these two vectors. This problem can be presented in mathematics as follows:

$$objective : \max \sqrt[t]{\sum_{i=1}^{r} | x_i - y_i |^t}$$

constraints :

$$\sum x_i = 1 \quad (i = 1...r)$$

$$\sum y_i = 1 \quad (i = 1...r)$$

$$x_i \geq 0, y_i \geq 0$$

Proof:

$$| x_i - y_i | \leq 1$$

$$| x_i - y_i | \leq | x_i + y_i |$$

$$| x_i - y_i |^t \leq | x_i - y_i | \leq | x_i + y_i |$$

$$\sqrt[t]{\sum | x_i - y_i |^t} \leq \sqrt[t]{\sum | x_i + y_i |} = \sqrt[t]{2}$$

Therefore, the range of the Minkowski distance between these two vectors is $[0, \sqrt[t]{2}]$.

A.3 Clack's distance

The Clack's distance is defined as:

$$\text{distance} = \sqrt{\sum_{i=1}^{r} \frac{| x_i - y_i |^2}{| x_i + y_i |^2}}$$

r : the number of dimension

x_i : ith dimension in vector x

y_i : ith dimension in vector y

Support these x_i, y_i in *r*-dimensional vectors *x*, *y* are probabilities. We want to calculate the range of the Clack's distance between these two vectors. This problem can be presented in mathematics as follows:

$$objective:\max\sqrt{\sum_{i=1}^{r}\frac{|x_i-y_i|^2}{|x_i+y_i|^2}}$$

$$\text{constraints}:$$

$$\sum x_i = 1 \quad (i=1...r)$$

$$\sum y_i = 1 \quad (i=1...r)$$

$$x_i \geq 0, y_i \geq 0$$

Proof:

$$\Theta\ (x_i-y_i)^2 \leq (x_i+y_i)^2$$

$$\therefore \frac{(x_i-y_i)^2}{(x_i+y_i)^2} \leq 1$$

$$\therefore \sqrt{\sum\frac{(x_i-y_i)^2}{(x_i+y_i)^2}} \leq \sqrt{\sum 1} = \sqrt{r}$$

Therefore, the range of the Clack's distance between these two vectors is $[0,\sqrt{r}]$.

A.4 Cosine distance

The Cosine distance is based on vector properties in a Euclidean space. It measures the Cosine angle in an *r*-dimensional vector space. This metric is defined as:

$$\text{distance} = \frac{\sum_{i=1}^{r} x_i \cdot y_i}{\sqrt{(\sum_{i=1}^{r} x_i^2)\cdot(\sum_{i=1}^{r} y_i^2)}}$$

r : the number of dimension

x_i : ith dimension in vector x

y_i : ith dimension in vector y

Support these x_i, y_i in *r*-dimensional vectors *x*, *y* are probabilities. We want to calculate the range of the Cosine distance between these two vectors. This problem can be presented in mathematics as follows:

$$objective:\max\frac{\sum_{i=1}^{r} x_i \cdot y_i}{\sqrt{(\sum_{i=1}^{r} x_i^2)\cdot(\sum_{i=1}^{r} y_i^2)}}$$

$$\text{constraints}:$$

$$\sum x_i = 1 \quad (i=1...r)$$

$$\sum y_i = 1 \quad (i=1...r)$$

$$x_i \geq 0, y_i \geq 0$$

Proof:

According to the definition, we have

$$\cos\theta = \text{distance} = \frac{\vec{x}\cdot\vec{y}}{|\vec{x}|\cdot|\vec{y}|} = \frac{\sum x_i \cdot y_i}{\sqrt{\sum x_i^2}\cdot\sqrt{\sum y_i^2}}$$

$$\therefore \frac{\sum x_i \cdot y_i}{\sqrt{\sum x_i^2}\cdot\sqrt{\sum y_i^2}} \le 1$$

Therefore, the range of the Cosine distance between these two vectors is $[0,1]$.

B Probability that trust value is less than basic trust value

We calculate the probability that the trust value is less than basic trust value. Several different mathematical approximations will also be presented. We keep all the assumptions in the simulations. And we only study the probability for car wash 1 in Section 6.2 Single service provider simulation.

We assume that the entity "Bob" has observed the data set $D=\{X_1=x_1, \ldots, X_n=x_n\}$ for which the *sufficient statistics* are $N=\{n_g,n_b\}$, where n_g is the number of times X="*good*" in D which consists of n independent identically distributed random outcomes, in other words, X is a Bernoulli trial with probability p. We know that n_g is a Binomial distribution.

$$P(n_g = k) = \binom{n}{k} p^k (1-p)^{n-k}$$

where the parameter p is the performance. The trust value can be calculated as follows:

$$trustValue = \frac{\alpha_g + n_g}{\alpha + n} = \frac{1+n_g}{2+n}$$

The probability that the trust value is less than the basic trust value is

$$
\begin{aligned}
&P(trustValue < basicTrust) \\
&= P(\frac{\alpha_g + n_g}{\alpha + n} < basicTrust) \\
&= P(\alpha_g + n_g < basicTrust \times (\alpha + n)) \\
&= P(\alpha_g + n_g < \alpha_g + basicTrust \times n) \\
&= P(n_g < basicTrust \times n) \\
&= \sum_{j=0}^{[basicTrust \times n]} P(n_g = j) \\
&= F_{n_g}(basicTrust \times n)
\end{aligned}
$$

where [*basicTrust*×*n*] is the largest integer less than *basicTrust*×*n*, Note if *basicTrust*×*n* is an integer, then [*basicTrust*×*n*]= *basicTrust*×*n* -1.

B.1 Approximation

1. Beta function

For more detailed information, please refer to the mathworld website at http://mathworld.wolfram.com/BinomialDistribution.html. Here we only present the results.

Let *k*=[*basicTrust*×*n*]

$$
\begin{aligned}
&P(trustValue < basicTrust) \\
&= \sum_{j=0}^{k} P(n_g = j) \\
&= P(n_g \le k) \\
&= 1 - \sum_{j=k+1}^{n} P(n_g = j) \\
&= 1 - I_p(k+1, n-k) = 1 - \frac{B(p; k+1, n-k)}{B(k+1, n-k)}
\end{aligned}
$$

where *B(p;k*+1*,n-k)* is the incomplete beta function and *B(k+1,n-k)* is the beta function.

2. Normal distribution

We present an approximation to the Bernoulli distribution for large *n*. For more detailed information, please refer to the mathworld website at http://mathworld.wolfram.com/BernoulliDistribution.html. Here we only present the results.

Define $\sigma^2 \equiv np(1-p)$ which is the variance of the Bernoulli distribution. When n is large enough, we get the following approximation (Demoivre-Laplace Theorem)

$P(n_g = k) \approx \frac{1}{\sigma\sqrt{2\pi}} \exp[-\frac{(k-np)^2}{2\sigma^2}]$. Note this is a normal distribution. We then can use the standard normal distribution to estimate the probability.

$$P(n_g < k) = P(\frac{n_g - np}{\sigma} < \frac{k-np}{\sigma})$$

$$\cong P(Z < \frac{k-np}{\sigma})$$

3. Poisson distribution

For p<<1, a different approximation procedure shows that the binomial distribution approaches the Poisson distribution. For more detailed information, please refer to the mathworld website at http://mathworld.wolfram.com/PoissonDistribution.html. Here we only present the results.

Let $\lambda = np$

$P(n_g = k) = \frac{\lambda^k e^{-\lambda}}{k!}$ Note that the sample size n has completely dropped out of the probability function, which has the same functional form for all values of λ.

For illustration purpose, the approximation is plotted in the following Figure 47 (refer to Appendix D.4 for MATLAB code).

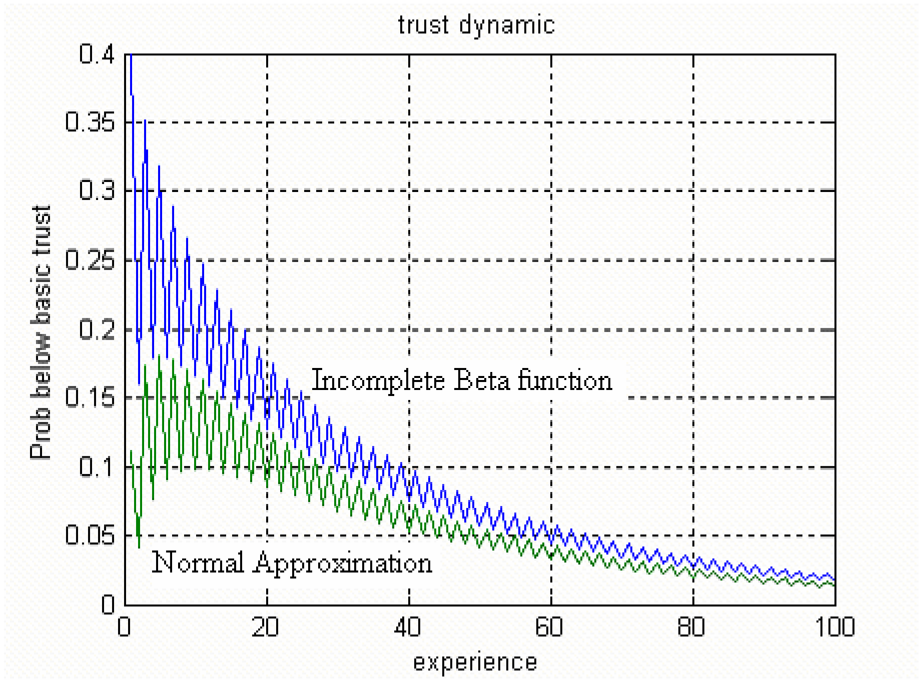

Figure 47: Approximation

C Range of experience number

According to experience number update relations, we can calculate the largest possible n as followings.

$n_k = n_k \times \gamma + 1$

$n_i = n_i \times \gamma$ *for* $i=1,\ldots r$ *and* $i \neq k$

$n = n_k + \sum n_i = n_k \times \gamma + 1 + \sum n_i \times \gamma = (n_k + \sum n_i) \times \gamma + 1 = n \times \gamma + 1$

$n = n \times \gamma + 1$

$n = 1/(1-\gamma)$.

When $\gamma=0$, the largest n is just 1, in other words, the entity forgets all the past experiences but the latest one. When $\gamma=1$, the largest n is infinite, in other words, the entity can remember all the past experiences.

D MATLAB code

D.1 Beta distributions

This MATLAB code is used to generate the beta distribution.

```
figure;
subplot(1,3,1);
apha1=12;
apha2=12;
theta=0:0.01:1;
p=gamma(apha1+apha2)./gamma(apha1)./gamma(apha2).*theta.^(apha1-1).*(1-theta).^(apha2-1);
plot(theta,p);
title('beta(p|12,12)');
xlabel('probability');
ylabel('pdf');
```

```
subplot(1,3,2);
apha1=2;
apha2=8;
theta=0:0.01:1;
p=gamma(apha1+apha2)./gamma(apha1)./gamma(apha2).*theta.^(apha1-1).*(1-theta).^(apha2-1);
plot(theta,p);
title('beta(p|2,8)');
xlabel('probability');
%ylabel('pdf');

subplot(1,3,3);
apha1=5;
apha2=3;
theta=0:0.01:1;
p=gamma(apha1+apha2)./gamma(apha1)./gamma(apha2).*theta.^(apha1-1).*(1-theta).^(apha2-1);
plot(theta,p);
title('beta(p|5,3)');
xlabel('probability');
```

D.2 Bayesian estimation

This MATLAB code is used to generate the Bayesian estimation 3D mesh.

```
clear all;
n=6;
[x,y]=meshgrid(0:1:20, 0:1:20);
z= x./(x+y);
mesh(z);
xlabel('α_g+g');
ylabel('α_b+b');
zlabel('probability');
gridon;
```

D.3 The upper bound of mistakes of WM

This MATLAB code is used to generate the upper bound of mistakes of WM.

```
clear all;
n=6;
m=20;
beta = 0.01:0.01:1
count=1;
for beta = 0.01:0.01:1
   mistake(count) = (log10(n) + m*log10(1/beta))/log10(2/(1+beta));
   count =count+1;
end;

i = 0.01:0.01:1
plot(i,mistake);
xlabel('beta');
ylabel('upper bound of mistakes made by WM');
grid on;
```

D.4 Probability that the trust value is less than basic trust value

This MATLAB code is used to generate the probability that the trust value is less than basic trust value.

```
clear all;
basicTrust = 0.5;
p = 0.6;
repeat=100;
for N= 1:repeat
   k = floor(basicTrust*N);
   if k == basicTrust*N
      k = k-1;
   end;
   sigma =sqrt( N*p*(1-p));
```

```
    z= (k-N*p)/sigma;
    ppp(N)=normcdf(z);
    pp(N) = 1-betainc(p,k+1, N-k);
end;

i=1:repeat;
plot(i,pp,i,ppp);
title('trust dynamic');
ylabel('Prob below basic trust');
xlabel('experience');
grid on;
```