Working Paper Series
ISSN 1177-777X

Trust-based recommendations for mobile tourists in TIP

Qiu Quan & Annika Hinze

Working Paper: 13/2008
December 8, 2008

Department of Computer Science
The University of Waikato
Private Bag 3105
Hamilton, New Zealand

Trust-based recommendations for mobile tourists in TIP

Qiu Quan, Annika Hinze
Department of Computer Science, University of Waikato
{qq6, hinze}@cs.waikato.ac.nz

Abstract *Recommender systems aim to suggest to users items they would like. However, concerns about the reliability of information from unknown recommenders influences user acceptance. In this paper, we analyse trust-based recommendations for the tourist information system TIP. We believe that the recommender strategy is closely related to the information domain applied. So, the delivered trust-based tourist recommendations have combined peers' ratings on sights, trust computations and geographical constraints. We create two trust propagation models to spread trust in the TIP community. Three Trust-based and location-aware filtering algorithms are implemented. According to research on feasibilities of trust in recommendation fields, three collaborative filtering algorithms in TIP are improved by introducing the trust concept.*

1 Introduction

The Mobile Tourist Information Provider (TIP) is designed to deliver dynamic and personalized tourist information to travellers. It is a combination of event notification techniques with location-based system and context-aware computing. The recommendation service in TIP provides information about touristic sites the users may want to visit.

The common techniques used in recommender systems are collaborative filtering and content-based filtering. Collaborative filtering tries to automate the "word of mouth" process on a world scale: the system recommends items that are liked by a group of similar users to an active user. Content-based filtering can suggest other items that belong to some categories preferred by the active user in the past. Although collaborative filtering is commonly used in some recommender systems to find like-mined users, it has some weaknesses such as the risk of malicious attrack, recommendations not being transparent to users, difficult control problem and so on. Content-based filtering simply recommends other items based on users' preferred subjects in the past. It also suffers a few weaknesses such as the new user problem and the restriction of the recommended items. A detailed evaluation of both algorithms is presented in *Section 2*.

In human society, interpersonal trust implies positive attitudinal similarity. As a result, recommendations given by trusted people usually have high acceptance and accuracy to the information user. Based on this, researchers have introduced trust into recommender systems in order to overcome several weaknesses cited before (See *Section 3*). Here, the trust is the explicit trust rating assigned to a peer. This information can be used to weave an on-line trust network for a user. The trust-based recommendation is generated from this personal community for an active user. Basically, generating trust-based recommendations involves two steps: the trust propagation and the trust recommendation generation.

We believe that the information domain has a considerable influence on recommendations. For example, the information taken for creating tourist recommendations is different with book recommendation creation. In travelling, tourists' preferences change frequently according to location. For the trust-based recommender in the TIP system, we believe that three factors have to be token into account in our trust-based recommendation construction: explicit ratings on sights, explicit ratings on peers and geographic coordinates of sights.

The goal of this paper is (1) the creation of trust propagation models for tourists, (2) the generation of trust-based and location-aware recommender algorithms, and (3) the evaluation of system functionalities and time efficiency of trust-based algorithms. Conducting of a survey to investigate tourists' responses of trust and trust-based recommendations during real travels.

In Section 2, we evaluate existing recommender strategies. Section 3 gives an overview of trust in human society and online applications. In Section 4, we new trust measures and algorithms – this is the core of this paper. Section 5 briefly presents the implementation environment and architecture of the trust-based recommender system. Section 6 discusses our evaluation results. Section 7 relates to our research of existing mobile tourist information applications. Finally, we summarize our findings in Section 8.

2 Background

This section clarifies the significance of recommender systems in the information society. Then, we introduce six criteria, which are measurements normally utilized to examine the performance of recommender systems. Subsequently, we present two basic information filtering strategies used in recommender systems: content-based filtering and collaborative filtering, and assess the performance of each one using the six criteria. Finally, we describe the environment of this project which is the TIP system, briefly introduce the history of the TIP system, system architecture and existing services.

2.1 Recommender Systems

We are overwhelmed by the data in the world, moreover, there is a large quantity of new information available every day in the internet society. It causes an information overload problem to arise. The information overload problem causes people to experience difficulties in locating truly useful information. A lot of effort has been invested into solving this problem, such as search engines, data mining techniques and recommender systems. Search engines let users search many kinds of information on-line by themselves. However, users always complain that the returned information is not what they want, because each user might use different keywords to search the same thing and the picked keywords are not matched with the keywords predefined in the search engine and so on. For these reasons, some information might not be discovered by users. Data mining techniques can intelligently predict results, but their slowness is a major drawback. Currently, these cannot be used to provide real-time recommendations. On the other hand, recommender systems aim to suggest only worthwhile information to users.

Recommender systems have been developed in a variety of forms. Usually, they appear as intelligent virtual assistants in a variety of E-commerce domains. By combining ideas of a user's information, information filtering and data mining algorithms, recommender systems have proven that they can deliver more intelligent and proactive information to users and match the preferences learned from users.

Typically, people would like to act upon recommendations from other people. For example, we go to see a movie because it is strongly recommended by reviews in the newspaper, or we listen to a new song because it is suggested by our friends. Our decision about acting upon recommendations is based on three premises [21]: (1) We trust the recommenders, (2) We assume that the recommender has sufficient knowledge of our tastes or tastes of people like us, and (3) We assume that the recommender has knowledge of the alternatives available. By taking a recommendation, we get a shortcut to things without having to try many things that we dislike, or having to have all knowledge about them. Fortunately, recommender systems automate this process by generating recommendations for users. For this reason, recommender system have become a popular service on-line.

In general, two basic information filtering schemas have been used in recommender systems: content-based filtering and collaborative filtering.

A content-based recommender [19, 32] relies on a plentiful supply of content representations of the items that are being recommended. For example, a content-based news recommender will rely on the keywords appearing in the articles, and the classification of the news. When the user requests news recommendations, the system needs to learn the user's preferred news categories from the user's historic reading materials. The system can then recommend other news from the user's preferred categories. Obviously, a content-based recommender needs a knowledge-learning engine inside the system in order to extract the relevant features from items. However, for some domains it is extremely hard to extract meaningful features from items, such as the genre of a song. To solve this problem, items must be manually tagged, a process which is expensive and error-prone.

The recommender system based on collaborative filtering is a possible alternative solution. This technique attempts to automate the "word of mouth" phenomenon on a large scale. The system finds a set of similar users to the information requester, and recommends items preferred by similar users to the requester. In contrast to the content-based filtering, collaborative filtering does not need to have representations of items, but it does need an additional rating system to capture and store users' ratings. For a target requester, recommendations are generated by gathering the items attached with high ratings from a set of similar users.

Trust is a very human and social concept, it is produced during people's exchange. Trust-based recommender systems aim to improve the performance of recommender systems by using trust-based filtering algorithms or enhancing recommender algorithms using trust.

2.2 Six evaluation criteria

Before looking at three recommendation strategies in detail (the collaborative filtering, the content-based filtering and the trust-based recommender system), we need to introduce six criteria commonly utilized to analyze the characteristics and performances of recommender approaches. The six criteria are (1) recommendation transparency, (2) the new-user problem, (3) computational complexity, (4) user control, (5) malicious attack resistance and (6) data sparseness. In this subsection, we give the definition for each of these criteria and explain the reason for choosing it as a measurement of the recommender system's effectiveness. Later on, these criteria are applied to evaluate content-based filtering, collaborative filtering, and the trust-based recommender.

1. *Recommendation transparency* is the degree to which the user knows who recommended certain items and why they were recommended.

   Typically, people are overwhelmed by information coming from different media. For example, commercial products are usually over glorified by the seller in order to boost sales. Normally, people do not want to be bothered too much by such information, or the information associated with unclear intentions. Thus, a recommendation, having a clear source and reason for the recommended item, is directly related to the user's acceptance.

2. *New-user problem* refers to the issue of offering recommendations to a new user who does not yet have any listed preferences.

   For some recommender approaches, the recommended results are partially generated based on the user's context (that might be the information about preferences pre-defined by users in their profiles, users' historic selections, etc). If the recommender system works on this principle, then a new user who utilizes the system for

the first time, might not be able to get any recommendations, because the user cannot supply any information to bootstrap the recommender algorithm. However, a friendly recommender system should have the flexibility to deal with this problem and still offer intelligent recommendations to the new user. For this reason, we utilize the "new-user problem" to evaluate whether the recommender system has the potential ability to handle this problem.

3. *Computational complexity* is the cost of computation when offering recommendations to all users. Computational complexity is closely related to the efficiency of the recommender algorithm. Having low computational load and a short response time is constantly required by on-line recommender systems.

4. *User control* refers to whether the recommender system is able to let the user interact with the internal algorithm in order to influence the recommended results.

   This becomes an essential issue when the user is not satisfied with the given recommendation. In addition, after the user has interacted with the system by adjusting some parameters, the recommender system has to perform well because the user has provided more precise preference information to the system. Furthermore, if the system lets the user control and correct recommendations, the user's confidence in the system can be built up.

5. *Malicious attack resistance* describes whether the recommended result can be easily manipulated by malicious users when they know about the principle of the recommendation algorithm.

   This measure relates to information security which is an important problem and need to be carefully considered by system designers, since we do not want our recommended results to be interrupted by malicious users.

6. *Data sparseness* is one of the major challenges for recommender systems. As users' preferences are diverse, the number of items that the user has chosen in the past only represents a small amount of the total items in the system.

   Because of lack of data, the recommender systems might not be able to produce sufficient recommendations, or not be able to produce any recommendations at all. For this reason, this measurement is used to examine the ability of the recommender system for handling sparse data.

In addition to the six measurements for analyzing the effectiveness of the recommender system, *recall* and *precision* are two other criteria applied to measure the accuracy of the recommendation on existing data sets. Recall is the percentage of relevant items that were returned, and precision gives the percentage of returned items that are relevant. These two criteria are normally used in information retrieval. To evaluate the quality of the recommendation, both measurements need to be performed off-line.

2.3 Analyzing content-based filtering

In this subsection, we start with a brief description of content-based filtering, and then the six criteria are utilized to examine the performance of the recommender system based on content-based filtering.

content-based filtering studies the user's historic selections in order to suggest items similar to the ones that the user liked in the past [21, 24]. To achieve that goal, a procedure for extracting the features of the items is needed. For some machine readable materials, such as news or books, their features can be automatically extracted by applying some algorithms (e.g. data mining algorithms). However, for images or movies, some meaningful features (e.g. genre) are difficult to extract by using current computer techniques. After finishing the feature discovery, the items need to be classified into categories according to their features. These categories are called semantic categories. Then, the content-based recommender can recommend other items in the same semantic category to the user. This category is the user's preferred category that the system has identified from the user's historic data.

1. *Recommendation transparency* The content-based algorithm performs well with regard to transparency. The user can be told about the source of recommendations, i.e. recommended items are other items in the same semantic groups that are liked by the user.

2. *New-user problem* This problem is the main weakness of the content-based algorithm. As we know, the users' historic data is the basis that content-based filtering works on. Consequently, because a new user lacks this historic data, pure content-based filtering is not able to produce any recommendations to the user.

3. *Computational complexity* The computational cost of the content-based algorithm is heavily dependent on the algorithm used to extract the features of the items. Normally, algorithms from artificial intelligence are applied to discover features from the machine readable items [19]. However, for some meaningful features, like the genre of a song, they still need to be determined and tagged manually. This process is expensive, error-prone, and highly subjective. However, if the features can be determined, it is then very cheap to filter the other items from the semantic categories to recommend to users.

4. *User control* From the principle of pure content-based filtering, we can see that the user cannot easily control the recommendation process, especially in the situation when the user comes to the system for the first time and receives no recommendation.

5. *Malicious attack resistance* content-based filtering cannot easily be attacked by malicious users, because content-based filtering concentrates on studying each user's information individually; it does not count social

relationships among users. As a result, we can safely say that content-based filtering is invulnerable to malicious attack.

6. *Data sparseness* As long as the user provides some historic data, the content-based recommender is able to produce recommendations to the user. But, the recommendations may concentrate on a few subjects only, if the user's historic data is sparse. The quality of the recommended result can still be quite high, it can be proved by comparing the user's preferred categories against the recommendations.

Overall, the recommendations generated by content-based filtering can reach a high accuracy. This can be shown by comparing the recommendations with the user's historic data. However, the recommended items are restricted to the subjects that the system has identified from the user's historic data. For this reason, content-based filtering may lack the ability to recommend items belonging to other categories that could potentially attract the user's attention.

The most difficult task for content-based filtering is the detection of item features. In addition, the new-user problem is also a significant difficulty. Nevertheless, content-based filtering presents transparency and good malicious attack resistance.

2.4 Analyzing collaborative filtering

collaborative filtering is automating the "word of mouth" process. It is considered to have great potential in recommender systems. We will examine collaborative filtering using the same six criteria as were used to judge content based systems.

collaborative filtering is also known as "social filtering" or "similarity-based filtering" [22,32]. Consider the following example. When we are about to go out for dinner, we often ask some friends with similar eating habits about the choice of the restaurant and then we act based on their suggestions. Collaborative filtering tries to automate this process. Instead of the users asking known people, the system (that knows the judgement of everyone) finds similar users to the requester and recommends the items liked by those similar users.

A collaborative filtering recommender system needs to have a rating system in which each user is asked to give her/his explicit options for selected items in terms of a numeric rating. In order to create recommendations for the current user (an active user), the standard collaborative filtering recommender takes three steps: (1) It compares the active user's ratings against every other user's rating (n-dimensional space). Usually, the similarity measure used is the *Pearson correlation coefficient*. (2) Based on the ratings of the most similar users, ratings are predicted for those items that the active user has not yet rated. (3) The items with the highest predicted ratings are the recommendations given to the active user.

1. *Recommendation transparency* For the collaborative filtering algorithm, the process of creating recommendations is not fully transparent to the user. Although the user might have been told that the recommended items are liked by similar people, the user has not been informed about the similarity measure and who gave the recommendations.

2. *New-user problem* collaborative filtering utilizes the overlapped information of ratings between other users and the active user. If the active user has not yet rated any items, purely collaborative filtering is not capable of supplying any recommendations to the user.

3. *Computational complexity* Computational complexity is one of the major weaknesses of collaborative filtering. If every user is considered as a vector of ratings on items, grouping all users together results in a rating matrix, whose dimensions are given by the product of the number of users by the number of items. According to the description of the principle of collaborative filtering, we can see it is a lazy, instance based learning algorithm, and the computational complexity grows linearly with the number of users and items. In order to find the most similar user, collaborative filtering needs to compare the active user with every other user in the system. If there are in total A users in the system, there will be $(A-1)$ similarity computations taking place for each active user. As a result, when the recommender system offers recommendations to all users, the computational complexity is $O(|A|^2)$ [19]. This means that standard collaborative filtering cannot scale to large environments with millions of users and items.

   Accordingly, because of the scalability problem, the collaborative recommender system may work in a centralized environment, but it is certainly not suitable for a decentralized environment.

4. *User control* collaborative filtering works like a black box to users. It is very difficult or impossible for users to control the recommending process when they are not satisfied with the created recommendations. So, if the recommender system gives bad quality recommendations, typically the user just stops using it.

5. *Malicious attack resistance* collaborative filtering can be easily attacked by malicious users. The simplest attack is the copy-profile attack: If the attacker knows information about a target user, the attacker can copy the information of the target user to create a fake user having exactly the same context as the current user. This fake user will be regarded as the most similar user. If the attacker makes a set of fake users by copying, these fake users must have large influence on the recommendations provided to a legitimate user.

6. *Data sparseness* Data sparseness is another difficult problem that collaborative filtering has to face, because it relies on information overlap. If none of the other users' ratings overlaps with the active user, in other words, no similar users can be found from the system, standard collaborative filtering cannot produce any recommendations.

The recommendations based on collaborative filtering are generated according to the opinions of similar users. The advantage of collaborative filtering over content-based filtering is that collaborative filtering does not need to know the features of the items, and collaborative filtering can predict new and potentially interesting items to users from other similar users. Collaborative filtering is simple and effective. However it has several weaknesses: it is computationally expensive, easily attackable and cannot cope with data sparseness.

2.5 The TIP system

The Tourist Information Provider [7, 12] (TIP) is designed to provide context-sensitive travel information from a variety of services to travellers on their travelling route. We develop our recommender strategies in the context of teh TIP project.

The system is implemented as a client-server architecture, supporting both desktop computers (for travel planning) and mobile device clients (for information delivery on-the-go). A number of versions have been developed as the TIP project progresses – the work reported in this paper uses TIP version 2.5. On the server side, the TIP system has a database back-end using a PostgreSQL database, which saves all users' information and all information about tourism, with PostGIS extensions for storing and searching the spatial data. The JAVA content of TIP is Apache's Jakarta Struts framework. The Struts framework is a Model-View-Control (MVC) architecture pattern. By using three separated modules, the programmer is able to reuse proven solutions and focus on developing robust solutions to new problems. Apache Tomcat is used as the web servlet container for the TIP.

On the client side, a web browser is used for displaying travel information and is responsible for the interaction between users and the TIP system. The client can be thin or thick depending on the services requested by the user. For example, the map service requires a thick client, while the recommender service requires only a thin client. For more details see [8, 9].

The recommendation service supplies further interesting sights to the user, taking into account the user's context and the sight's context [10]. Three recommendation components have been implemented by utilizing the user's known preferences and the current context of both user and sights [11].

Our goal is to add trust-based recommendation into the recommendation service. To generate trust-based recommendation, the trust-based recommender needs to fetch data from the data layer and the geographical data supplied by the GPS location service. Subsequently, the trust-based algorithms are applied to create recommendations and display them on the web page.

3 Trust problems and challenges

At the beginning of this section, we will clarify the trust concept followed by an explanation as to why trust can solve the problems existing in recommender systems (discussed in *Section 2*). After that, we give a description of trust in TIP. Then the problems and challenges of Trust-based recommendations in TIP are presented. At the end of the section, we outline the focuses of this paper.

3.1 Trust concept

This subsection briefly presents the research on the trust concept. Then we describe people's attitude toward recommendations coming from different sources (information systems and real friends). Based on this research, we clarify our motivation for studying trust in the recommender system. At the end of the subsection, we give a few examples of application systems that have employed trust in their system for different goals.

3.2 Research on the trust concept

In the real world, trust is produced during people's exchange [28], and trust is subconsciously hidden behind people's decisions. So, it is a very human and social concept, and has been intensively studied by researchers in different fields (marketing, anthropology, sociology and philosophy, etc) including computer science. Here, we prefer to use Gambetta's definition of trust: *"trust (or, symmetrically, distrust) is a particular level of the subjective probability with which an agent will perform a particular action, both before [we] can monitor such action (or independently of his capacity of ever being able to monitor it) and in a context in which it affects [our] own action"* [6]

Riegelsberger has conducted detailed research on the trust topic [28]. He discussed the trust concept from a social and psychological point of view, and presented positive consequences that trust can create. For example, trust can be useful for both parties when they are engaging in exchanges, and can reduce the cost of transactions. Instead of the traditional face-to-face communication, new on-line technology has become a novel media for people's interaction. It has been observed that supporting trust and trustworthy techniques can make the technology-mediated interaction go smoothly. E-commerce is one of the examples, and it can only run successfully by providing users with a high level of trust interaction.

3.3 Recommendations by friends preferred

By taking recommendations, we can have a shortcut to things we like without having to try many things we dislike or without having to acquire all knowledge about them. Typically, information systems can automate this procedure to generate recommendations for users. Here, we are interested in the quality of recommendations provided by information systems.

After analyzing six on-line recommender systems, three book recommender systems and three movie recommender systems, Sinha and Swearingen gave the following conclusion in their paper [30]: Users prefer receiving recommendations from people (e.g. friends and family members) they know and trust well to an on-line recommender system. The reason is that recommender systems only have limited, domain-specific

knowledge about users. However, friends know the user, and hold sufficient knowledge about the user's tastes in a number of domains. Their suggestions can be easily acknowledged by the user. But the user is also interested in items recommended by online recommender systems, because they are able to offer "new" and "unexpected" items, while friends might recommend items matching previously identified interests. This report presents the concrete reason for studying the trust concept in the real world and the need for improving the performance of recommender systems using trust.

3.4 Motivations of trust

We have seen the great significance of trust. Here, we present three points about trust that motivate us to study it in recommender systems.

1. *Correlation between trust and user similarity* Although everyone has a particular definition of trust in her/his mind, people naturally grouped by trust always hold some positive attitudinal similarities [19]. Abdul-Rahman and Haile [2] have presented positive research results about trust and user similarity. They concluded that in a certain community, for example communities of people playing tennis, their members start to build ties of friendship and trust if they have similar interests among them. This phenomenon reflects some social reasons in the real world, such as people's need to have social affiliation. Because of the positive correlation between trust and user similarity, it is meaningful to integrate the trust concept into recommender systems.

   Recommendations from like-minded people sharing similar interests have significant meaning for the information requester. However, past research [1] has shown that recommender systems based on user similarity are not suitable for a large environment with millions of users and items. Because of the correlation between trust and user similarity, it is possible that the efficiency of similarity-related algorithms can be further improved by introducing the trust concept.

2. *The weaknesses of content-based filtering and collaborative filtering motivate the trust-based recommender system*

   content-based filtering needs to have a representation of the items in terms of features. However, extracting meaningful features from items is a difficult procedure. While a trust-based recommender system only needs information provided by users (explicitly stated trust in other users and ratings on items), it is totally independent of the features of the items.

   collaborative filtering recommender systems do not take into account social relationships in the environment. We believe that the direct judgement of users on users (trust) can enhance the performance of recommender systems by providing more reliable and accurate recommendations and also be invulnerable to malicious users.

3. *Trust can be propagated* Trust has some inherent transitivity properties. For example, if A trusts B, and B trusts C, A might trust C as well, even though they do not know each other. If we propagate trust in the social network, a small number of trust statements can cover a large portion of the trust network. For this reason, indirectly trustable peers are very useful information resources to boost the system when the number of direct trustable friends is too small to provide sufficient recommendations.

3.5 Trust-related applications

Trust information has been employed in a variety of information systems. Although every system utilizes the trust concept for a different goal, the common thing is to introduce a sort of social control over the system. For example, on-line communication software, weblogs, E-commence, peer-to-peer (P2P) decentralized systems.

Online communication software, e.g., MSN (messenger.msn.com), allows users to contact their friends around the world at any time. Trust can be seen as the concept of users permitting other users to connect to them, and blocking contact from distrusted users. MSN helps users to build personal communities, Hi5 (`www.hi5.com`) is the software associated with MSN. It utilizes the users' friend list (trustable peers) in MSN to extend the independent personal community to a social community network. In other words, they weave a huge global social network using explicit trust among people. As we know, one of the difficult problems in E-business is how to attract more users to register with an on-line system. Hi5 implemented a fast and cheap way to absorb as many MSN users as possible to build a global social network. Later, these users became potential costumers of their business.

Weblogs are a kind of on-line diary that are frequently updated web sites, and are easy to create and maintain. Because of the simplicity of personal web creation, there are millions of people in the world who have their own weblogs. Links between weblogs and items support the decentralized construction of a rich information network (called blogoshere). Some weblogs have expressed other kinds of information within XML files. For example, in a fofa.xml (Friend-OF-A-Friend) file, users can state who are friends and who they can trust. By navigation of fofa.xml, users can visit other on-line diaries through their trustable friends.

Ebay.com is an example of an E-commerce application system. It is a web site that provides a service for on-line auctions. Ebay constructs a virtual community for its users. After every commercial transaction, the trader is asked to rate their partner. Then the reputation of every user as a seller or buyer is computed and displayed to users. In the ebay community, every user can behave freely. The rating expressed by one user of other users contributes to building users' reputation in the system. Users can approximately foresee the safety and success of having trade with an unknown buyer or seller according to their reputation.

Currently, most on-line application systems use reputation, such as ebay and Amazon. Reputation is a concept that is similar to trust, but slightly different. Trust involves two peers. Each peer holds a particular trust value about the other, which means each user will gain different trust values from different users. Reputation is a property of a user assigned by the embedded social community [19]. It means each user in the system will be granted a global value of reputation by the system. Reputation only can be applied in the centralized system.

P2P is a decentralized system. It is a class of applications that takes advantage of resources, such as storage, cycles, content, human presence, which are available at the edges of the internet [29]. Here, the trust concept is used to distinguish trustable equipment on the internet to avoid the risk of viral or other attacks.

3.6 Trust can solve the problems

We are now going to use the same six criteria we defined in *Section 2* to examine recommender systems based on trust, and explain how trust-awareness can help to overcome problems in recommender systems. We believe that taking into account direct judgements of users on users can improve the performance of recommender systems.

1. *Recommendation Transparency* Lack of transparency is one of the main weaknesses of the recommender system based on collaborative filtering. Users like to understand why particular items were recommended. The recommendations generated by the complicated similarity measure cannot provide clear and understandable reasons about recommended items to users. In addition, the most similar users might not be known by the active user. For this reason, it is hard for users to follow recommendations.

   Trust metrics operate on a naturally extended social network. In addition, it has been proved that interpersonal attraction has a close correlation with attitudinal similarity. As a result, recommendations provided by the user's community (naturally grouped by trust) are transparent to the user. It is possible to let the user track back to find who recommended items and why they recommended them.

2. *New-user problem* The new-user problem is another major disadvantage of both collaborative filtering and content-based filtering. Especially for the recommender system based on user similarity, because no information overlap can be found between a new user and the other users in the system, the recommender system cannot produce any recommendations.

   However, in a social environment, if a new user has some trustworthy friends, these explicitly stated friends can let the recommendation process function correctly. Moveover, trust has inherent transitive properties, a small number of trust statements can cover a big proportion of the social network if we propagate trust. Following trust propagation, it is possible to predict trust of indirectly connected friends. Both directly and indirectly trusted peers are reliable information resources helping to create recommendations.

3. *Computational complexity* In general, recommender systems require heavy computation. For the collaborative filtering algorithm, it needs to compute user similarity (normally the *Pearson correlation* is applied) offline rather than on-the-fly, because of the scalability problem. Certainly this approach is not suitable for a decentralized environment.

   The positive correlation between trust and user similarity can reduce or even eliminate the computational complexity. Although the direct known peer group only includes a small number of users in the system, the extended peer group formed in trust propagation might include dissimilar users, and the degradation of similarity is not as fast as the increase of the members in the group, when supposing that trust does correlate with user similarity [32]. Moreover, according to our survey, people would like to accept suggestions from dissimilar peers if they have built strong trust relationships.

4. *User control* As discussed in *section 2*, recommender systems based on collaborative filtering or content-based filtering act as a block box to the user. Users only receive recommendations, but they do not know how they were generated. In particular, they cannot control the recommendation process.

   Allowing users to control and correct recommendations is required when users are not satisfied with the recommended items. In order to achieve this, the recommender system must expose some internal recommendation processes to users. In the trust-aware recommender system, this can be easily achieved. For example, displaying the trust statements of some users lets users correct them, and users are allowed to include or exclude peers from their peer groups.

5. *Malicious attack resistance* In collaborative filtering, every user in the system is taken into account in the same way. In this case, there is no way to discover the malicious user. As a result, attackers can easily influence the created recommendations if they know how the recommender algorithm works.

   While the trust metric can be attack-resistant, this is achieved by only taking into account recommendations of "reliable" users, both directly and indirectly trusted friends. This process not only can easily exclude malicious users and fake users from the personal on-line community, but also can reduce the user base and the rating base involved in the computation of recommendations.

6. *Data sparseness* In a trust-based recommender, if the data sparseness is due to the fact the user has not

yet defined the directly trusted peer group, that trust-based recommender cannot produce any recommendations from the peer group. Otherwise, the problem of data sparseness is heavily related to the trust-based recommender strategies used. For example, if the trust-based recommendation is generated from a set of similar friends (apply collaborative filtering on the active user's social community), it might cause no recommendation if there is no information overlap between the active user and the peers. However, if the recommendation is generated from trusted friends only (apply the trust-based filtering), the rating history of the active user is not counted in the recommendation. As a result, data sparseness in the user's ratings cannot affect the recommending process.

Table 1 gives a summary of the performance regarding the three recommender strategies on the six criteria. It can be seen that the trust-based recommender is superior compared to the pure content-based filtering or the pure collaborative filtering on almost all measures. Typically, the trust-based recommender shows clear recommendation transparency and low computational complexity, provided the possibility of user control and malicious attack resistance. For the problem of data sparseness, the performance of the trust-based recommender depends on inside recommender strategies used.

This result is our initial motivation to utilize the trust concept to enhance the recommender components in the TIP system. Our goal is not only to generate personalized recommendations to meet the needs of each individual user, but also to increase accuracy and user acceptance of the results.

3.7 Trust in TIP

We have described the advantages of introducing the trust concept into recommender systems. In this subsection, we propose a way to use trust used in the Trust-based Recommendation system for the TIP. We believe that trust has a close relationship with the information domain. In the tourist information domain, we believe that trust has to be added with additional domain information: geographic information.

The goal of the TIP system is to dynamically deliver tourist information to travellers in their travel routes. The Trust-based Recommendation system for TIP aims to provide accurate, reliable and acceptable tourist information to a particular traveller according to the travel history of the peer group and the traveller's current location. Before discussing detailed information about our trust models and trust-based algorithms for TIP, we need to introduce our types of trust used for constructing trust models. Here, four kinds of trust are worthwhile to be introduced: local trust, reputation and two domain-related trusts: geographic trust and location-aware reputation.

Local trust (personal trust and person-group confidence trust) In this work, local trust includes personal trust and the person-group confidence trust. Personal trust is a trust statement representing a particular level of the subjective trust of the one on the other. Some users can be trusted or distrusted by someone else. Here, personal trust is explicitly estimated by an active user to one for her/his friends according to the friend's visited sights and the ratings the friend gave to the sights. In general, the active user only has direct judgements about a small group of users who are direct friends. Each user may have his/her own personal trust rating from different users. Because the system allows users to explicitly state their trust in their friends, the active user can reach some unknown peers through trust propagation. So it is possible to predict trustworthiness of unknown users. For example, if A trusts B and B trusts C, it is possible to compute how much A trusts C.

Person-group confidence trust is a set of trust statements directly assigned to a direct peer on a number of sight groups. Compared to personal trust, the person-group confidence trust reflects more precise trust information on each perspective of the source peer to the target peer.

Reputation Reputation is a global trust metric approximately computed by the community as a whole for a specific user [19]. Basically, a user's reputation is the average of all trusts received from other users. As a user's reputation is closely examined by all other users in the system, users having a high reputation can be regarded as having sufficient domain knowledge of the recommended items, and their advice can benefit other users.

Compared to local trust, reputation is more objective. A high reputation means a user's knowledge or advice is recognized by other users. By using reputation, users are provided an alternative choice when they want to look for recommendations from domain-experienced (high reputation) users. On the other hand, introducing reputation into the system can encourage users to behave well. In other words, it lets users be responsible for their behaviour in an on-line society. Although the credibility of reputation might not be compatible with personal trust (because reputation is given by the system, which might not be recognized by the individual), the proportion of user coverage is higher than for personal trust. Usually it is close to 1.

In general, the local trust can achieve a higher accuracy than reputation (global trust), because it is more precise and belongs to a single user, while reputation reflects the overall judgement of the whole community for a specific user. But sometimes, a single user does not agree with the judgement of the whole community. Comparing the computational cost of local trust against reputation, it can be found that the computation of local trust is more expensive since the system must calculate trust for every single user, while the system only needs to run the algorithm once to obtain reputations for all users.

Domain-related trust As we know, recommender systems act like intelligent assistants to suggest only worthwhile information in a certain information domain to the information requester. Obviously, different domains have their own particular characteristics. For this reason, we need to carefully analyze the domain features and requirements before apply-

	RT	NUP	CC	UC	MAR	DS
content Filtering	+	−	+(−)	−	++	+
collaborative Filtering	−	−	−	−	−	−
Trust-based recommender	++	+	++	++	++	+(−)

Table 1: Comparison of three recommender strategies regarding our six criteria RT: Recommendation transparency; NUP: New-user problem; CC: Computational Complexity; UC: User control; MAR: Malicious attack resistance; DS: Data sparseness Symbols: "++" means very good; "+" means good; "+(−)" means somewhat; "−" means poor.

ing a system design.

Our system goal is to recommend tourist information. The key feature of tourist information is that each place or sight is always associated with a piece of precise location information. For example, the "University of Waikato" is located at "1 Knighton road, Hamilton, New Zealand". We might not need to consider geographic difference when we recommend books, but we have to carefully consider geographic information for each sight when we recommend tourist attractions. For example, if a Chinese is on a trip in New Zealand, suggestions from Chinese friends, who have not been in New Zealand, might not be useful to the traveller. Instead, the traveller might like to get recommendations from other travellers who are currently in New Zealand, or the local residents, even through they are not known to the traveller.

We adapt trust by adding geographic information to create two types of domain-related trust: geographic trust and location-aware reputation.

1. *Geographic trust* Geographic information has significant meaning in the real world. For example, what is the distance between us and the nearest bookshop? And also in travelling, we assume geographically near tourists are more likely to hold up-to-date travelling information. For instance, some tourists may know that the open time of the museum has changed because of a social event, or a flower show has been cancelled because of bad weather, or some tourists may have found a new interesting place they wanted to suggest other tourists might like to visit. For these reasons, geographically closer tourists are an important information resource for the current traveller. If we draw a circle around the active user, all tourists in this region are regarded as geographically close users to the active user. From the trust point of view, we believe that the shorter the distance between the active user and another user, the higher the geographical trust the second user will get from the first.

2. *Location-aware reputation* As discussed above, reputation is a global trust value approximately calculated by the whole community for a particular user. For travellers, we add geographic information into reputation to derive a location-aware reputation. To find other travellers with high location-aware reputations for an active user, we need to identify a physical region according to the location of the current traveller. For this active user, the location-aware reputation travellers are the users who have visited this area, and whose reputations (average of the received personal trust) are in the list of the top N users.

3.8 Challenges for TIP

Before presenting the design of trust propagation models and trust-based recommender algorithms in detail, we need to briefly summarize problems that we have identified.

1. *Store and display the trust relation of the TIP community* The trust relationship of users in an on-line community is the basis for weaving a social network and propagating trust. Different system environments require different schemas for storing and fetching data. In centralized environments, the relational database is a suitable tool for effectively saving and querying the data. In the decentralized environment, normally a semantic file is responsible for linking and exchanging information among entities in the network. The semantic file contains information about itself and its trusted neighbours.

   Presenting the social network to users is a challenge. A visualized community can help users know how their on-line communities are constructed, and it also can help users understand recommendations. For this reason, we need to implement some visualizing interfaces to present the trust community to the single user.

2. Predict trust for an unknown user Trust is inherently transitive. To precisely compute a trust for an indirectly unknown peer, we need to study the possible ways of propagating trust in the real world. This problem is related to a deep understanding of trust as well as distrust in the real community. Typically, researchers have not intensively studied distrusted peers or behaviors online. We think that studying distrusted behaviors in the on-line community can help to distinguish malicious users and faked data.

3. *Explanation of recommendations* When we present recommendations, we need to carefully consider giving a proper presentation of the recommended results. For example, a approach of displaying results on a mobile device for which the screen is too small to show much information. Information regarding recommendations includes: the recommended items, computed ratings for them, categories that recommended items belong to, the number of users who recommended them. Besides, we may also need to explain the reasons and factors that

influence recommendations, present the users who have given recommendations and how they rated items and friends.

4. *Reputation calculation* Computing a global objective reputation for each user in an on-line community can be a challenge. Each user has different and subjective opinions about a particular user. The calculated reputation based on users' subjective assessments (personal trust issued by users to others) may not reflect the real reputation for a given user. We think that the user's reputation in the system needs to combine the received personal trust and the user's contributions. In the TIP community, the user's contribution may include the number of sights that the user has visited, the number of reviews that the user has given to sights, other users' feedback about the quality of recommendations, and so on.

5. *Improve recommending ability of the trust-based recommender system* In the trust-based recommender system, if the user has not yet defined any friends, the user may not get any recommendations. To fix this problem, instead of using local trust, location-aware reputation and geographic trust are alternative choices that can be used to provide recommendations for users.

6. *Problem of data privacy* Personal data privacy is a critical issue in an on-line system. Some users may not want to share their data with other users, even though these users have been labelled as friends or friends of a friend. For this reason, the system must have a service for protecting the data privacy of users. In a decentralized environment, this issue becomes vitally important as each entity in the network can freely behave as it likes, which relates to the prevention and reduction of the risks of potential attacks.

7. *Measure the quality of recommendations* For recommender systems, one of the challenges is the lack of objective and effective ways to evaluate the performance of a system. To evaluate the quality of recommendations, we can perform off-line testing on the data set. However, off-line measures cannot capture user's acceptance, while running a recommender evaluation with many real users to test the hypothesis is difficult in the research community. Moreover, the lack of real trust-based data sets is another problem for analyzing the effectiveness of the algorithm.

8. *Update the trust value* The trust value needs to be automatically updated according to satisfaction about recommended sights. In general, users do not change their settings frequently. So automatically adjusting the trust according to user satisfaction is needed for detecting users' preferences in order to improve the quality of recommendations later on. One simple way of evaluating user satisfaction is to examine feedback scores that users give to recommended sights. If users give high feedback scores to them, the system needs to automatically raise the trust value of the peer who gave the recommendations. Otherwise, the system will automatically take some trust off, if other users give unsatisfied recommendations.

9. *Decentralized trust-based recommendation for the TIP system* Real travellers may also require a decentralized TIP recommender system on which users can contact other unknown travellers in the travel route. For the decentralized trust-based tourist recommendation system, we need to consider several problems below:

   (a) A semantic file setup. The semantic format file is responsible for linking and exchanging information among entities in the distributed environment. For the decentralized trust-based recommender, this file need to contain the user's information, current location, trusted friends and trust ratings, communication protocol, etc.

   (b) A data structure for storing data on the local device and exchanging data between trustable travellers.

   (c) Identification of the geographic trustable travellers, and the ability to communicate with them.

   (d) Presentation of the geographically trustable users on an electronic map.

3.9 Methodology

We focus on using explicitly stated trust among users to create recommendations with high accuracy and user acceptance. We consider the following four aspects:

1. Creation of trust propagation models for predicting trust of unknown peers. We are interested in predicting the precise trust value of an unknown user that the active user does not know directly, but where there is a connection by a trust chain.

2. Design trust-based and location-aware filtering algorithms to recommend near sights by combining trust and location information with sight ratings of the peer group. And design of an approach to enhance the performance of collaborative filtering in TIP by integrating trust, user similarity and location information.

3. We believe that the additional information of recommendations is a very influential factor in increasing acceptance of users toward recommended results. In the recommendation presentation design, we integrate trust models into the recommended results. Using this presentation, we are able to explain trust-based recommendations to users, such as: How many users recommend a sight? What is the relationship between the active user and recommenders? What's level of trust of an indirectly trusted peer? What ratings did peers issue to a sight?

4. Our trust-based recommender strategies are going to be examined using three methods: evaluation on system functionalities, evaluation on response times of each recommender algorithm for carrying out a recommendation, and data coverage involved in the recommendation procedure. To examine designs of trust-based tourist recommenders and to investigate requirements of real travellers in order to improve the system functionalities later on, we conduct a survey to gather subjective opinions of real travellers.

4 Trust-based Recommendations

This section introduces our trust concept and trust-based recommendations. We describe our three trust propagation models. Finally, four approaches of trust-based recommendations for the TIP will be explained in detail.

4.1 Concepts and Definitions

Peer A peer p is a uniquely identifiable autonomous entity and able to request and expose some information. We use it to indicate a user. A user in a social community is called a peer among peers.

Peer group A peer group represents a tourist social community. The biggest peer group in TIP is the whole community $P = \{p_1, p_2, \ldots, p_n\}$, which contains all peers. In total, there are n (finite number) peers in it. For each peer, there is an individual peer group that is a sub set of P. Suppose, a peer p_i with $p_i \in P$, the corresponding peer group is then referred to as P_i, with $P_i \subseteq P$.

Personal trust In the real world, if someone is well known by other people, they each hold a different and direct judgement regarding trustworthiness about her/him in their minds. Here, personal trust T_p is used to describe this kind of trust relationship existing in the TIP community. Every peer is allowed to express a personal trust in the other. Personal trust is defined by four properties as shown below:

1. Typically, each peer only explicitly issues a personal trust score to a peer that is a direct acquaintance. This is to mimic assessments between people regarding interpersonal relationships in the real world. Consequently, the trust value is heavily dependent on the individual's opinion and evaluation; thus, it is subjective. The trust for an unknown peer needs to be computed by the trust propagation model.

2. Personal trust statement T_p is a real value between 0 and 1. For the whole peer set P, the personal trust T_p can be expressed as:

$$T_p : P \times P \rightarrow [0, 1]. \quad (1)$$

   In our trust model, $T_p(p_a, p_b) = 1.0$ with $p_a, p_b \in P$ means the source peer p_a believes that the target peer p_b is the most entrusted peer. This might be because the source peer knows that they have similar hobbies or interests.

   Special attention should put on trust statement 0 ($T_p(p_a, p_b) = 0$); it might have two different possible judgements. The first one is that source peer p_a might trust target peer p_b very well, but they have totally opposite interests; the second one is that the source peer considers the target peer as a malicious user, and this target peer will be excluded from the peer group later on. In these two cases, the source peer issues the lowest trust to the target peer to indicate that all information coming from these two kinds of peer will be ignored. The peers outside the peer group always hold a trust value that is $null$.

   Each peer in a peer group P_i has a given trust statement which is a positive value and equal or greater than 0. Thus, for all peers in the peer group P_i holds:

$$p_j \in P_i \quad iff \quad T_p(p_i, p_j) \geq 0; \\ i, j \in [1, n], p_i, p_j \in P.$$

   Different to our approach, Levien's Advogato [17] trust metric only makes *Boolean* decisions regarding trustworthiness. It directly classifies the local groups into entrusted or distrusted ones.

3. Personal trust is not symmetrically distributed between two peers. For example, the source peer p_a trusts the target peer p_b in a certain level. This does not mean that p_b trusts p_a to the same level (so it may be that $T_p(p_a, p_b) \neq T_p(p_b, p_a)$). Perhaps p_b might not trust p_a at all. This property is different from the similarity between two peers. Similarity is one of main metrics used in collaborative filtering, it is symmetrically distributed. Two different users share one value of the similarity.

4. Personal trust is a customized and subjective value, which is subconsciously made, based on the quality of perceived information. For this reason, it should be easily defined and manipulated by the users themselves.

Reputational trust Reputation T_r is a global trust value for each user in the system. The reputation of peer p can be calculated by averaging the received personal trust values from the other users. So, the reputation is computed by the embedded community and as a property attached with each user. Because the reputation is closely examined by users, it presents the synthesized opinion of all other users to a particular user and can be regarded as an objective assessment. Corresponding to the personal trust (local trust), we call the reputation as reputational trust (global trust).

$$T_r : P \rightarrow [0, 1] \quad (2)$$

Domain-related trust As discussed in *section 3*, trust can associate with some domain information, when it applies to a particular domain. Different domains put different additional information on trust. In the tourist information domain, we propose two types of the domain-related trust: geographic trust and location-aware reputation.

1. Geographic trust The geographic trust T_g is the trust calculated based on the geographic distance between users. Each user has attached location information which is her/his geographic coordinates. The distance between two users can be calculated by applying the standard distance function. The threshold λ defines the longest distance between the source peer and the target peer. If the distance between the source peer and the target peer is less than or equal to λ, the target peer is given a geographic trust by the source peer. In other words, the target peer is a geographical trustable peer of the source peer. The geographic trust statement can be formalized in a geographical function below:

$$T_g : P \times P \rightarrow [0, 1] \tag{3}$$

$$T_{g_{(p_i,p_j)}} = 1 - \frac{distance(p_i, p_j)}{\lambda}; \tag{4}$$
$$i, j \in [1, n]; p_i, p_j \in P; distance(p_i, p_j) \leq \lambda$$

where p_i is the source peer, and p_j is the target peer. The domain of the geographic trust is [0,1], where 1 means the target peer is at same location with the source peer and 0 means the distance between the source peer and the target peer exceeds the maximum distance threshold λ. The target peer who is closer to the source peer has gained a high value of geographic trust, because her/his domain knowledge can benefit the source peer more.

2. Location-aware reputation The location-aware reputation T_{lr} is similar to reputation, but it contains additional location information. According to the location of the active user, we can define a region for her/him. For this active peer, the other users who hold location-aware reputation, must have visited this area. Because they have similar travel experiences, their suggestions can benefit the active user.

$$T_{lr} : P \times P \rightarrow [0, 1] \tag{5}$$

Trust threshold μ The trust threshold μ is defined by users self. This measure is used to identify individual trustable peers. All peers in the user's peer group, their trust values must satisfy with the trust threshold. This parameter is designed to prevent involving too many peers into the peer group when propagating trust on the trust network.

Trust path A trust path graph ρ (see Figure 1) [5] consists of two finite sets: a vertex set $V(\rho) \subseteq P$ (where each vertex represents a peer or a user) and a directed edge set $E(\rho)$ (where each directed edge is associated with an ordered pair of vertices). If edge e is associated with the ordered vertex pair (a, b), then e is said to be the directed edge from a to b. The personal trust value is the weight of the edge issued by a to b. If one vertex is reachable from the other, a trust path must exist between them. A trust path is a finite sequence of adjacent edges connected via vertices. Thus, a trust path can be described as the list of vertices or peers:

$$\rho[p_i, p_j] = p_i - p_k - \ldots - p_j; \quad p_i, p_k, \ldots p_j \in P \tag{6}$$

where p_i represents the source peer and p_j represents the target peer.

To prevent a trust cycle existing on the trust path, we define that there is no repeated vertex (peer) on one path, all vertices (peers) on one path are unique.

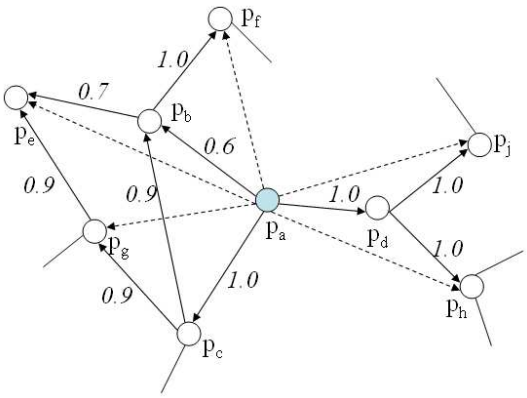

Figure 1: Trust path graph

Consequently, the personal trust can only be propagated along the trust path if the peer has at least one friend. Otherwise, there is only one isolated vertex (the user him/herself) on the path. Because of the transitivity of trust, a peer is able to reach a directly unknown peer through the trust path.

Direct/indirect friends Direct friends of the source peer are all adjacent peers on the trust paths which are connected by exactly one directed edge. The remaining peers on the same trust path are indirect peers of the source peer.

Length of the trust path The number of the steps s is used to measure the length of a trust path. The number of steps between two peers on one trust path is the number of the edges between two vertices (peers). For example (see Figure 6), the number of steps of the trust path $\rho[A, B]$=$A - B - C - D$ is $s = 3$.

Trust decay As described above, the trust relationship is transitive via the directed trust path. However, as the trust path of the source peer to the target peer becomes longer, the trust degree should decrease gradually. For example, if p_b is trusted by p_a and p_e is trusted by p_b, it might follow that p_a might trust p_e as well, although p_a does not know p_e directly. But the indirect trust attitude from p_a to p_e should be lower than the direct transitive trust attitude from p_a to p_e. We will call this behavior *trust decay*. The domain of trust decay is between 0 and 1. 1 means the trust is 100% transmitted to the

peer. 0 means there is no trust transmitted to the peer. For all peers in the system, the trust decay D is defined as:

$$D : P \times P \rightarrow [0, 1]. \quad (7)$$

We specify that the trust decay from the source peer to each direct friend is 1. When a third peer is added to the end of the trust path, the trust decay from the source peer to the third one will be $(1-d)$. The value $d \in [0, 1]$ is the *decay constant*. Accordingly, the trust decay for each further step is decreasing simultaneously, as the trust path spreads further. For this reason, the trust decay is closely related to the number of steps between two peers on the trust path. The expression of the trust decay between two peers can be formulated as:

$$D(p_i, p_j) = (1 - d)^{s-1}, \quad p_i, p_j \in P, d \in [0, 1]. \quad (8)$$

where s is the number of steps between the source peer p_i and the target peer p_j on a trust path ρ.

Another reason for utilizing the trust decay is to avoid the worst case occurring on the trust path. For example, there might be 100 peers on the trust path. If each peer issues 1 (100% trust) to the neighbor peer, the calculated trust of the source peer to every target peer will be 1 without considering trust decay. In fact, it cannot represent the real trust tendency among people in the community. By integrating trust decay into the trust computation, this worst case can be avoided.

Moreover, introducing the trust decay allows us to let closer friends have more influence on the recommendations (their recommendations are close to the top of the recommendation list), because their suggestions are likely to be appreciated by the source peer.

Trust network The community trust network graph N is constructed by connecting direct trust paths of all peers together. Each individual peer has one personal trust network graph N_i, which is a partial network of N. One personal trust network N_i consists of all directed trust paths from the source peer p_i to every other peer in the peer group. By expanding the trust network, the source peer can connect to unknown users via the peers.

Figure 2 shows an example of the personal trust network of peer p_a. In this case, the source peer p_a has three direct friends, p_b, p_c and p_d, respectively. Both p_c and p_d have other two direct friends, and p_b has three. The solid edges indicate that two peers have a direct trust relationship (they are direct friends). They are associated with explicit trust scores. The dashed line means that two peers are indirectly connected (they are indirect friends). The trust of the source peer to the indirectly connected target peer can be predicted along with the trust path. Because there may be more than one path that can be identified between the source peer and the target peer from the trust network. For example, p_a is able to connect to p_g through three different paths: $\rho[p_a, p_g]_1 = p_a - p_b - p_g$, $\rho[p_a, p_g]_2 = p_a - p_c - p_b - p_g$ and $\rho[p_a, p_g]_3 = p_a - p_c - p_g$. Hence, there are two issues derived, one is trust path selection, the other is trustworthiness prediction. These issues are discussed in *subsection 4.2*.

Sight The sight set $S = \{s_1, s_2, \ldots, s_n\}$ contains all uniquely identifiable sights of the system. Each sight $s_i \in S$ is a place where the traveller might like to visit.

Feedback The feedback f is a value expressing the subjective assessment the peer has in the sight. It is reflected how much the peer likes the item. Each feedback statement is represented as a numeric value ranging from 0 and 10. The feedback can also be seen as the subjective knowledge contributed by the peer.

Sight group Sight group set $G = \{g_1, g_2, \ldots, g_k\}$ contains all classifications of sights. Each sight group $g_i \in G$ represents one specific category that a sight $s_i \in S$ might belong to. Sight groups are constructed in a hierarchy. One sight can belong to several sight groups. All sights in one sight group are sharing a similar topic. The sight group set θ ($\theta \subseteq G$) is the immediate sight categories of all sights.

Location Location is the geographical coordinate regarding the physical location of the sight or the user.

Nearby sight group A nearby sight group S_λ, $S_\lambda \subseteq S$, contains sights that are geographically close to a given location. λ is the distance threshold utilized for constraining nearby sights.

Trust-based recommendation Corresponding to our domain of recommendation, generated trust-based recommendation R_{p_i} for a particular user p_i is an ordered top N strongly recommended nearby sights that have been associated with computed scores and a list of trustable peers (recommenders):

$$R_{p_i} = \{(s_{\lambda_1}, f_1, P_{i_1}), (s_{\lambda_2}, f_2, P_{i_2}), \ldots, (s_{\lambda n}, f_n, P_{i_n})\}, \quad (9)$$

where $s_{\lambda_n} \in S_\lambda$ is one of the nearby sights, f_n is the computed score given to this sight. $P_{i_n} \subseteq P_i$ is the set of peers that recommended this sight.

4.2 Trust propagation model

In the previous subsection, we have given seventeen definitions. Here, these definitions are applied to describe our three trust propagation models designed for the trust-based recommendation in TIP.

4.2.1 Trust model 1: Propagating Boolean trust

This trust model is the implementation of the most well-known local group trust metric: Levien's Advogato metric [32]. We let every user issue the Boolean decision (0 or 1) of trustworthiness to the direct peers. The trust will be propagated if the personal trust of the peer is 1, otherwise the trust spreading will stop. For example, if A trusts B and B trusts C ($BooleanTrust(A, B) = 1; BooleanTrust(B, C) = 1$), trust will propagate from A to C. If A trusts B and B does not trust C ($BooleanTrust(A, B) = 1; BooleanTrust(B, C) = 0$), the trust will propagate from A

to B and stop at B. All peers in one peer group are having the same trust value: 1.

4.2.2 Trust model 2: Propagating numeric trust

Different from the first solution, we let each user issue a numerical trust value to a direct friend. In this way, a user can express her/his level of trust to other users. Therefore the web of trust of all users can be aggregated in a global trust network. This network does not have a central control.

Because users are allowed to cast trust values to other direct peers, using these explicit numeric trust statements, it is possible to precisely predict trust in an unknown peer by propagating trust. For example, A issues the trust statement in B with value 0.7, and B issues the trust statement in C with value 0.8, it is possible to calculate how much A could trust C.

To predict trust in an unknown peer, a graph working algorithm is used. This algorithm involves two steps: the trust calculation and the trust path selection.

1. *Calculation of the trust on a given path*

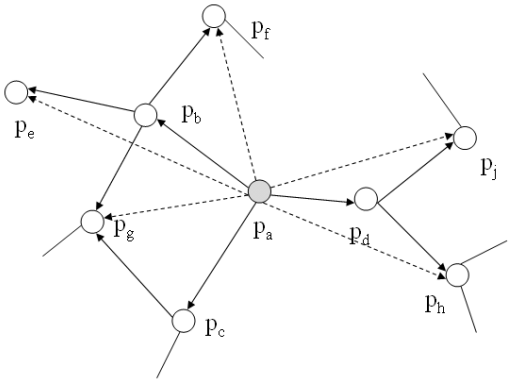

Figure 2: p_a's trust network graph

In order to calculate the trust value T of the source peer to a target peer on a given path ρ, three steps are necessary.

(a) Find the personal trust value that is associated with each directed edge on the trust path. The propagated personal trust value of the source peer to the target peer is the product of all personal trust values attached to the edges on a path between them. The computed personal trust value can be expressed as:

$$T_p(p_{speer}, p_{tpeer}) = \quad (10)$$
$$T_p(p_{speer}, p_i) * T_p(p_i, p_j) * \ldots * T_p(p_n, p_{tpeer}) \quad (11)$$

where $T_p(p_i, p_j)$ is the personal trust value issued by p_i to p_j, and p_{speer} is the source peer; p_{tpeer} is the target peer.

(b) Compute the trust decay from the source peer to the target peer according to the number of edges on the path between two peers (see Formula 8).

(c) The trust value T from the source peer to the target peer is gained by multiplying the propagated personal trust T_p with the trust decay D.

$$T(p_{speer}, p_{tpeer}) = T_p(p_{speer}, p_{tpeer}) * D[p_{speer}, p_{tpeer}] \quad (12)$$

For example, Figure 3 shows a directed path graph from p_a to p_e; two directly connected edges are included in it. Each edge is associated with a direct trust value. So the personal trust from p_a to p_e is $T_p(p_a, p_e) = T_p(p_a, p_b) * T_p(p_b, p_e) = 0.9 \times 0.7 = 0.63$, and the trust decay of p_a to p_e is with Equation 8: $D(p_a, p_e) = (1-d)^1$, if we set the decay constant $d = 0.1$, $D(p_a, p_e) = (1-0.1)^1 = 0.9$, the final computed trust from p_a to p_b is $T(p_a, p_e) = T_p(p_a, p_e) * D(p_a, p_e) = 0.567$.

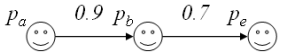

Figure 3: Trust transition 1

2. *Principle of choosing the trust path* If there are several paths existing between the source peer and the target peer, we only select one of them to propagate the trust.

In our trust model, to avoid trust cycles on a trust path, we define that the path for propagating trust only contains induplicated peers. The principle of the trust path selection is to choose the trust path from a set of pathes between the source peer and the target peer with over all maximum trust flow. To achieve it, first of all we need to identify all trust paths of the source peer to the target peer from the trust network. Then we need to compute the trust on each path (Formula 12). Finally, the path with the maximum trust value is picked as the trust path from the source peer to the target peer, and this value is the final trust between them. The resulting formula for computing the trust is:

$$T(p_i, p_j) = \quad Max \forall \rho[p_i, p_j] \ \{\prod_{k=1}^{w} T_p(p_i, p_j)(1-d)^{s-1}\}, \quad (13)$$

where w is the number of trust paths that connect the source peer and the target peer, $\forall \rho[p_i, p_j]$ includes all w paths between p_i and p_j, $p_i, p_j \in P$, and $p_j \in P_i$.

There are a few solutions to deal with the same problem. Massa and Bhattacharjee [21] chose the minimum number of steps needed to reach every other user. They believe that the users next to the current user on the trust network are predicted as more trustable than users further away. Abdul-Rahman and Hailes [1] average the trust value over the paths in their trust model.

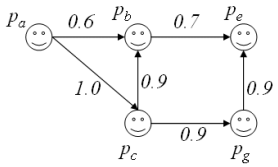

Figure 4: Trust transition 2

Here, we give an example of how to calculate trust. Figure 4 shows a trust network. Three paths can be identified from p_a to p_e. According to the description above, the trust rating of p_a to p_e through the trust path $\rho[p_a, p_e]_1 = p_a - p_b - p_e$ is: $T_{\rho[p_a,p_e]_1} = 0.6 \times 0.7 \times (1-0.1)^0 = 0.378$. The trust through the path $\rho[p_a, p_e]_2 = p_a - p_c - p_g - p_e$ is: $T_{\rho[p_a,p_e]_2} = 1.0 \times 0.9 \times 0.9 \times (1-0.1)^2 = 0.6561$. And the trust through the path $\rho[p_a, p_e]_3 = p_a - p_c - p_b - p_e$ is

$T_{\rho[p_a,p_e]_3} = 1.0 \times 0.9 \times 0.7 \times (1-0.1)^2 = 0.5103.$

As a result, the second trust path ($\rho[p_a, p_e]_2 = p_a - p_c - p_g - p_e$) has got the highest trust flow. Thus, the trust of p_a to p_e is 0.6561.

4.2.3 Trust model 3: Propagating the trust person-group confidence

This trust model is a modified version based of the second trust model. In this model, we not only propagate the personal trust information, but also propagate the person-group confidence trust along with the trust path.

Definition: Person-group confidence Person-group confidence is a numeric value in $[0, 1]$ that illustrates how much a peer trusts his friend on a particular aspect (which is a sight group in TIP). This value implies the user's preferred subject. In the real world, people's preferences are different, and this also holds for the two most trusted friends. For example, a traveller likes to go to beaches and visit museums (user's preference). He knows that one of his friends is good at surfing and less interested in museums (peer's preference). It is easy to imagine that this friend might give high scores on surfing places, while assigning low scores to museums. If the traveller asks for recommendations from this friend on those two subjects, he might have a high confidence in the recommendations on the beach category and low confidence in the recommendations of the museum category (user's judgement of the recommended sights).

In this paper, we refer to the personal confidence regarding the number of sight groups ($\theta = \{g_1, g_2, \ldots, g_m\}$, the immediate sight categories of sights) of the friend as a set of *person-group confidence*. It is displayed as a numeric vector.

$$C_\theta^{p_i}(p_i, p_j) = \{c_{g1}^{p_i}(p_i, p_j), c_{g2}^{p_i}(p_i, p_j), \ldots, c_{g_m}^{p_i}(p_i, p_j)\} \quad (14)$$

Where $p_i, p_j \in P$, $p_j \in P_i$ and $g_m \in \theta$. Each element in the vector is representing a confidence value on a particular sight group and holds a real value in $[0, 1]$. Similar to personal trust, the person-group confidence needs to be explicitly issued and modified by a user to direct acquaintances. The person-group confidence is a function:

$$C_\theta^P : P \times P \times \theta \rightarrow [0, 1] \quad (15)$$

In the trust-based recommender for TIP, each single user is asked to issue a person-group confidence vector to a direct friend. If there are a number of friends in the user's peer group, the user will hold a person-group confidence matrix of friends. We refer to the matrix for user p_i as $C_{Matrix_{p_i}}$ The person-group confidence matrix of a user p_i can be expressed as:

$$C_{Matrix_{p_i}} : P \times \theta \rightarrow [0, 1]. \quad (16)$$

By using the person-group confidence, users can precisely express their preferences. According to this information, the system can focus on recommending user preferred sight categories. Although, users are required to provide more information to the system, a usability study has shown that users do not mind providing more inputs to the recommendation system, if they can get better recommendations [17].

Propagating the person-group confidence on the trust network Similar to propagating trust, it is possible to predict the person-group confidence vector in an unknown peer. Propagating the person-group confidence on the trust network needs to take three steps. The first step is to select the trust path with the maximum trust flow between source peer p_1 to target peer p_n (Formula 13). The second step is to get the person-group confidence vector issued by p_{n-1} to p_n. Finally, the person-group confidence vector is updated using the maximum trust flow. The resulting vector is the computed *trust person-group confidence vector* of the source peer to the target peer.

The trust person-group confidence is the combination of the person-group confidence and trust. The formulated function for it is shown below:

$$C_\theta : P \times P \times \theta \rightarrow [0, 1]. \quad (17)$$

$$C_\theta(p_i, p_j) = C_\theta^{p_i}(p_i, p_j) T(p_i, p_j), \quad p_i, p_j \in P$$

Table 2 is an example showing how to compute the trust person-group confidence of p_a to p_d from Figure 1. Here, we take $A - B - C$ as the propagated trust path of p_a to p_c and set the decay constant $d = 0.1$. In this case, p_c has visited four sight groups. Peer p_c is a indirect friend of p_a and a direct friend of p_b. The propagated trust value from p_a to p_c is 0.56. The trust decay from p_a to p_c is 0.9. According to p_b's individual preference and judgement, the person-group confidence issued by p_b to p_c on four different subjects is $\{1.0, 0.9, 0.2, 0.5\}$. The resulting trust person-group confidence vector from p_a to p_c is $\{0.504, 0.4536, 0.1008, 0.252\}$.

To sum up, we have proposed three trust propagation models. The first one is very simple and straightforward. Users issue the Boolean value (0 or 1) to the others, and trust will not propagate if the trust value is 0. All peers in peer groups are 100 percent trusted by users. The second model considers the trust as a numeric value. In this case, users are able to express the level of trust in their friends. For an unknown peer, a numeric trust is predicted in the period of trust

Sight group	Person-group confidence of p_b to p_c	Personal trust of p_a to p_c	Trust decay of p_a to p_c	Resulting confidence of p_a to p_c
g_1	$c^{p_b}_{g_1}(p_b, p_c) = 1.0$	$T_p(p_a, p_d) = T_p(p_a, p_b)T_p(p_b, p_c) = 0.9 \times 0.8 = 0.56$	$D(p_a, p_c) = (1 - 0.1)^1 = 0.9$	$c^{p_a}_{g_1}(p_a, p_c) = 0.504$
g_2	$c^{p_b}_{g_2}(p_b, p_c) = 0.9$			$c^{p_a}_{g_2}(p_a, p_c) = 0.4536$
g_3	$c^{p_b}_{g_3}(p_b, p_c) = 0.2$			$c^{p_a}_{g_3}(p_a, p_c) = 0.1008$
g_4	$c^{p_b}_{g_4}(p_b, p_c) = 0.5$			$c^{p_a}_{g_4}(p_a, p_c) = 0.252$

Table 2: Example of the trust person-group confidence from p_a to p_d

propagation. The third model not only propagates the personal trust, but also propagates the trust confidence on each sight group. We believe the third model transfers more precise information of the user's judgement and preference.

4.3 Trust-based Recommendation generation

In this subsection, we will present four approaches of the trust-based recommendation. In every single description, we will use enhanced notations with logical operators: *The construct ON event IF condition DO action*, will be used to describe each approach. This notation is inspired by active database triggers and has also been used in [13].

The notations given in Table 4.3 are used to refer to various data sources. Table 4.3 shows the methods used in the following descriptions.

Variable	Description
S	Set of all sights
S_λ	Set of nearby sights
P	Set of all peers
P_i	Peer group of user p_i
P_i^s	Recommenders of the sight s, who are in the user's (p_i) peer group
H	Set of all historic data of peers
H_{p_i}	Set of historic data of a user's (p_i) peer group
λ	The distance threshold for finding near sights
F	Set of all feedbacks of all sights
C_{p_i}	Set of all confidence values
T_{p_i}	Set of all trust values
COR	Set of correlation coefficients of the user against peers
d	The trust decay of the source peer to the target peer
R_{p_i}	Set of recommendations

Table 3: Notations used in this paper

Defining the nearby-sight group In the real world, when a traveller arrives at a city or a sight, only surrounding sights are meaningful and useful to the traveller. In addition, for travel planning, a user might similarly require recommendations close to a given location. In both cases, a set of sights near to a given location has to be determined.

To find the near sights, we need to get the geographic coordinates of the location. A GPS device on the mobile device or an electronic map is able to provide precise geographic information. By using it, the neighboring sights can be filtered out from the sight table: The distance between a nearby sight and the input location must be less than or equal to a predefined distance criterion λ. The set of nearby sights is $S_\lambda = \{s_{\lambda 1}, s_{\lambda 2}, \ldots, s_{\lambda l}\}$ with $S_\lambda \subseteq S$. A simple location event le_1 occurs when a user p_1 is at location l_1.

ON location event $p_1.le_1$
IF $p_1.le_1.loc \in S \wedge \exists s \in S : distance(s.loc, p_1.le_1.loc) \leq \lambda$
DO $\forall s$ (near sights): $AddSight(s, S_\lambda)$

The recommended sights must be in the nearby sight set.

4.3.1 Generating trust-based and location-aware recommendations by propagating the Boolean trust value

This recommendation generator uses the first trust propagation model: propagating Boolean trust on the trust network. This approach involves three steps. The first step is trust propagation in which user p_1 can find a trust peer group P_1. The second step is to collect historic data set H_{p_1} of the peer group regarding all nearby sights. Finally, recommendations are generated based on the peers' historic data only.

Step 1: trust propagation A trust propagation event $t1_1$ occurs when user p_1 requests recommendations at location l_1. At the beginning, user's peer group P_1 only contains the user.

ON trust propagation event $p_1.t1_1$
IF $\exists p_i \in P \wedge p_1.t1_1.P_1 \subseteq P : DirectFriend((p \in P_1), p_i) \wedge p_i \notin P_1$
DO $\forall p_i$ (peers of p_1): $AddFriend(p_i, P_1)$

After finishing the even above, user p_1 has a peer group that contains all direct friends. If the user wants more entrusted friends, we need to expand the users' peer group by including indirect friends into the user's peer group.

Following the trust network, indirect and potentially trustable peers can be found. They are the direct friends of those peers that are already in the user's peer group. We define there is no duplicated peer in a peer group. After including the indirect friends into the peer group, the peer group will get extended from P_1 to $P_1^{'}$. It will then contain both direct friends and indirect friends. By recursively executing the procedure above, the user's peer group gets growth.

Method	Description
$distance(s.loc, p.loc)$	True if the sight is near to the user's location
$AddSight(s, S_\lambda)$	Add a *sight* into the nearby sight group S_λ
$DirectFriend(p_i, p_)$	True if peer p_j is the direct friend of user p_i, false otherwise
$USimilar(p, p_1)$	Compute the correlation coefficient between the source peer and the target peer
$AddFriend(p, P_i)$	Add peer p to peer group p_i
$AddTrust(t, T_{p_i})$	Add a trust t to the trust set of user p_i
$AddConfidence(c, C_{p_i})$	Add a confidence c to the confidence set of user p_i
$AddCorCoef(cor, COR)$	Add a correlation coefficient to the correlation coefficient set COR
$ComputeTrust(P_i, p)$	Compute the trust of the source peer to the target peer
$update(H, T_p, D)$	Update the historic data set by the given personal trust and trust decay
$update(H, C, T_p, D)$	Update the historic data set by the given confidence vector, personal trust and trust decay
$CollectHistoricData(h_p, H_{p_i})$	Add historic data of peer p into a user's historic data set H_{p_i}
$recommend(H_{p_i}, P_i)$	Recommend top N nearby sights to the user using trust-based filtering
$Fill(h_p)$	Fill on-feedback values in the rating matrix
$CollaborFilter(H_{p_i}, T_{p_i}, P_i)$	Generate recommendations using the TIP collaborative filtering algorithm
$TrustEnhanced(R_{p_i}, T_{p_i})$	Trust is used to enhance recommendations generated from collaborative filtering

Table 4: Methods used for recommendation generation

Step 2: collecting historic data Now the user's peer group contains a set of peers. From travel histories of peers, the sights that the peers have visited and that are also in the nearby sight group will be extracted. These sights are associated with feedbacks given by peers. The extracted data forms a peer's historic data set H_{p_1}. A data collection event c_1 occurs after finishing the trust propagation for user p_1.

ON data collection event $p_1.c_1$
IF $\exists p \in p_1.c_1.P_1 : \exists h(p) \in H \wedge \exists h(p).loc \in S_\lambda$
DO $\forall h(p)$ (historic ratings of peers):
$CollectHistoricData(h(p), H_{p_1})$

Step 3: generating recommendations Before generating recommendations from a historic data set H_{p_1}, the data in H_{p_1} needs to be aggregated. There are two reasons to do so. The first reason is that a peer might have visited a sight more than once. The second one is that one sight might have been visited by more than one friend. Thus, the final rating of the sight is the average of the feedback issued by all friends for the same sight. While aggregating, we store the number of peers who visited the same sight. Finally, the generated recommendation R_{p_1} from H_{p_1} is displayed as an ordered sight list, which contains top N strongly recommended nearby sights, the sets of peers who suggested the sights and the average scores to the sights.

A recommendation generation event r_1 occurs after completing the data collection.

ON recommendation generation event $p_1.r_1$
IF $p_1.r_1.H_{p_1} \neq null : p_1.r_1.P_1 \neq null$
DO $recommend(p_1.r_1.H_{p_1}, p_1.r_1.P_1)$

In this approach, the recommended sights from direct and indirect recommenders are treated equally. This idea conforms the hypnosis that naturally grouped people share similar tastes and interests.

4.3.2 Generating trust-based and location-aware recommendations by propagating the numeric trust value

Our first solution assumes that naturally grouped peers are sharing similar interests with the user, but it does not consider the trustworthiness of peers. Our second approach tries to integrate the level of trust into recommendation generation. This approach involves four steps. Again, the first step is still the trust propagation to get a peer group of the user. The second step is to collect the historic data set H_{p_i} from all peers. The third step is to integrate the trust into H_{p_i} to form a trust-based rating data set $H^t_{p_i}$. Finally, the recommended sights are generated based on $H^t_{p_i}$.

Step 1: trust propagation Analogous to the first approach, the peer group of user p_i can be found by propagating trust on the trust network. Then the trust value T, which is combination of personal trust T_p and trust decay D from the user to each peer, is calculated. The overall maximum trust flows are always chosen as the real trust values of the source peer to the target peers.

A trust propagation event $t2_1$ occurs when user p_1 requests recommendations at location l_1.

ON trust propagation event $p_1.t2_1$
IF $\exists p \in P : DirectFriend((p_i \in p_1.t2_1.P_1), p) \wedge p \notin P_1 : d = ComputeTrust((p_i \in p_1.t2_1.P_1), p) \rightarrow \exists t_p \in T_p \wedge (t_p.userid = p_i \wedge t_p.friend = p)$
DO $\forall t$ (trust values of peers), $\forall p$ (peers of p_1): $AddFriend(p, P_i)$, $AddTrust(t, T_{p_1})$

Step 2: collecting historic data The historic travelling data of peers is extracted to construct a data set H_{p_i}. A data collection event c_1 occurs after finishing the trust propagation for user p_1.

ON data collection event $p_1.c_1$
IF $\exists p \in p_1.c_1.P_1 : \exists h(p) \in H \wedge \exists h(p).loc \in S_\lambda$
DO $\forall h(p)$ (historic ratings of peers):
$CollectHistoricData(h(p), H_{p_1})$

Step 3: integrating trust into the historic data Now we use the trust matrix to update the data set H_{p_i} by multiplying the corresponding trust values with the ratings from peers. This results in a trust-based historic data set $H^t_{p_i}$. By now, the trust concept has been integrated into the peers' data. A data update event $up1_1$ occurs after finishing the data collection.

ON data update event $p_1.up1_1$
IF $p_1.up1_1.H_{p_1} \neq null : p_1.up1_1.T_{p_1} \neq null$
DO $\forall h(p)$ (historic ratings of peers): $update(H_{p_1}, T_{p_1})$

Step 4: generating recommendations Consequently, the recommended sights can be generated from $H^t_{p_i}$. Thus, the resulting recommendation R_{p_i} is also an ordered top N recommended sights.

A recommendation generation event r_1 occurs after completing the data update.

ON recommendation generation event $p_1.r_1$
IF $p_1.r_1.H^t_{p_1} \neq null : p_1.r_1.P_1 \neq null$
DO $recommend(p_1.r_1.H^t_{p_1}, p_1.r_1.P_1)$

In this approach, we allow the user to interact with the system by issuing numeric personal trust to each acquaintance. The explicitly issued personal trust contains the level of trust regarding the direct known recommenders. For indirect trust peers, their trust values are predicted by the trust propagation model. Subsequently, the historic data set is updated by trust. The updated historic data set can be seen as containing the user's individual preferences of recommenders. As a result, recommendations are generated from most trustable peers. By using the second trust model, users can interact with the recommender system through expanding or narrowing down the peer group and modifying the personal trust values to influence the recommendation process.

4.3.3 Generating trust-based and location-aware recommendations by propagating the trust person-group confidence

Our second solution added the factor of the trustworthiness of peers into the recommendation generation. However, trustworthiness is too coarse to precisely represent the real trust relationship between peers. In fact, one person only trusts someone else in some rather than in all aspects since each person's preferences are unique in the world. To represent this trust feature, a trust person-group confidence vector is used to specify this kind of trust. This is realized by our third approach.

In the third approach, four steps are needed to accomplish the recommendation process. Firstly, the peer group P_i is created for the user; secondly, the trust person-group confidence matrix $C_{Matrix_{p_i}}$ is computed; thirdly, the trust person-group confidence matrix is used to update the extracted historic data set from H_{p_i} to $H^c_{p_i}$; finally recommendations are generated from the confidence-matrix-based historic data set.

Step 1: trust propagation The initial peer group P_i of the current user still contains direct friends only. Then the trust of the source peer to each target peer is calculated. And the person-group confidence vector of the target peer is obtained from the trust path with the maximum trust flow. After that, a trust person-group confidence matrix $C_{Matrix_{p_i}}$ is constructed for the active user by grouping together trust person-group confidence vectors of all peers in the user's peer group.

A trust propagation event $t3_1$ occurs when user p_1 requests recommendations at location l_1.

ON trust propagation event $p_1.t3_1$
IF $\exists p \in P : DirectFriend((p_i \in p_1.t3_1.P_1), p) : d = ComputeTrust((p_i \in p_1.t3_1.P_1), p) \rightarrow \exists t_p \in T_p \wedge (t_p.user = (p_i \in p_1.t3_1.P_1) \wedge t_p.friend = p) \wedge \forall c \in C \wedge (c.user = (p_i \in p_1.t3_1.P_1) \wedge c.friend = p)$
DO $\forall p$ (peers of p_1),$\forall c$ (confidence values of peers), $\forall t$ (trust values of peers): $AddFriend(p, P_1)$, $AddConfidence(c, C_{p_1})$,$AddTrust(t, T_{p_1})$

Step 2: collecting historic data A data collection event c_1 occurs after finishing the trust propagation for user p_1.

ON data collection event $p_1.c_1$
IF $\exists p \in p_1.c_1.P_1 : \exists h(p) \in H \wedge \exists h(p).loc \in S_\lambda$
DO $\forall h(p)$ (historic ratings of peers):
$CollectHistoricData(h(p), H_{p_1})$

Step 3: integrating trust person-group confidence vectors into the historic data Update the historic data set H_{p_i} using the trust person-group confidence matrix to form a confidence-matrix-based historic data set $H^c_{p_i}$. Eventually, the recommendation can be generated from $H^c_{p_i}$. A data update event $up2_1$ occurs after finishing the data collection.

ON data update event $p_1.up2_1$
IF $p_1.up2_1.H_{p_1} \neq null : p_1.up2_1.T_{p_1} \neq null : p_1.up2_1.C_{p_1} \neq null$
DO $\forall h(p)$ (historic ratings of peers): $update(H_{p_1}, C_{p_1}, T_{p_1})$

Step 4: generating recommendations A recommendation generation event r_1 occurs after completing the data updating.

ON recommendation generation event $p_1.r_1$
IF $p_1.r_1.H^c_{p_1} \neq null : p_1.r_1.P_1 \neq null$
DO $recommend(p_1.r_1.H^c_{p_1}, p_1.r_1.P_1)$

Our third approach not only considers the personal trust, but also the trust confidence in different aspects. So, the updated historic data set contains the information about user preferred recommenders and the user preferred recommended subjects. Accordingly, the generated recommendations from the confidence-matrix-based historic data set should be closer to the user's interests and more easily accepted by the user. Because of considering different trusts on different sight groups, the computational cost in the recommending process is much higher than in the second approach.

4.3.4 Enhancing the collaborative filtering by trust and location-aware

As stated in *section 3*, people naturally grouped by trust always hold some positive attitudinal similarity. This is the motivation for us to exploit explicitly stated trust to enhance the accuracy and user acceptance of recommendations created by collaborative filtering.

The basic idea of trust-enhanced collaborative filtering is to let the source peer increase the trust rating in the target peer in order to increase the influence of opinions from the target peer when forming recommendations. Pure collaborative filtering is based on the similarity between the source peer and the target peer. In our case, we define that if A trusts B, the similarity between A and B can be computed. The combination of the trust statement and user similarity implies how much a user trusts a similar peer.

We believe that trust and user similarity can complement each other. In general, users explicitly issue trust to their peers according to previously received information from them. User similarity is computed by the algorithms based on the input rating statements explicitly provided by users. Combining trust and user similarity together can deal with some specific situations, such as, "the peer fully trusts the other one, but they are totally different", or "although two peers are similar, they do not trust each other". For the second case, we need to deeply understand the meaning of such social situations. It might be the case that a malicious user has copied the ratings from the other in order to influence recommendations.

The recommendation of trust-enhanced collaborative filtering is a list of top N appreciated near sights generated from most trustable and similar peers. We believe that users' satisfaction toward recommendations of directly trusted friends is high because users directly express their trust to their peers, while the satisfaction resulting from user similarity and indirectly trusted friends is relatively lower because the user similarity and the indirect trust are computed by algorithms.

The forth approach simply includes four steps to generate recommendations. The first step is the trust propagation; the second step is user similarity computation; the third step is to predict ratings for the active user; in the final step, the trust matrix is used to weight predicted ratings of recommended sights, then produce a ordered top N recommendation list.

Step 1: trust propagation A trust propagation event $t2_1$ occurs when user p_1 requests recommendations at location l_1.

ON trust propagation event $p_1.t2_1$
IF $\exists p \in P : DirectFriend((p_i \in p_1.t2_1.P_1), p) \wedge p \notin P_1 : d = ComputeTrust((p_i \in p_1.t2_1.P_1), p) \rightarrow \exists t_p \in T_p \wedge (t_p.userid = p_i \wedge t_p.friend = p)$
DO $\forall p$ (peers of p_1), $\forall t$ (trust values of peers): $AddFriend(p, P_1)$, $AddTrust(t, T_{p_1})$

Step 2: user similarity calculation The rating statements of each peer are taken from current user's peer group P_i as input of user similarity metric, and the output is the similarity value of current user p_i against the peer. In this step, the *Pearson Correlation coefficient metric* is performed, which has proved to be the most efficient algorithm to compute user similarity in the collaborative filtering. The Correlation coefficient is between $+1$ and -1.

The most common strategy is to use positive similarities only because users with a negative correlation are dissimilar to current user. However, in the real world, people usually have some friends with totally different interests. They are regarded as dissimilar users according to the computed correlation, even though people are still interested in suggestions of them because their information is reliable, especially the tourist information based on our survey. Moreover, because of location restriction and small coverage of the peer group, the data sparseness of the sight rating matrix that is collected from peers is more serious. As the reliability of friends has been closely examined by users, simply disregarding the information of dissimilar but trustable friends might lose valuable data source. For this reason, we consider to use information of both positive and negative similar peers to generate recommendations.

As discussed before, collaborative filtering relies on information overlap. In this case, every user is considered as a vector of ratings on items, grouping users together results in a matrix, whose dimensions are given by the product of number of users by number of sights ($users \times sights$). Because each user only rated some rather than all sights, there is no-feedback score to the remaining sights. As the resulting

rating matrix is sparseness, so the main task of collaborative filtering is to reduce data sparseness. In the collaborative filtering of the TIP system, no-feedback values are replaced by either a high score (10) or a neutral score (5) based on three assumptions below.

1. Using information of the user profile It is assumed that the user is likely to issue a high feedback score to a sight, if the sight is belonging to the sight category that is the user preferred and has explicitly defined in the profile. Based on this assumption, a score of 10 is allocated to a sight which belongs to a sight group preferred by the user, or a neutral score of 5 otherwise. Thus, the sparseness of rating matrix is demolished.

2. Using information of travel history For some reasons, users might not assign any score to visited sights. In this case, we check user travel history and make a assumption that a user likes a particular sight if she/he has visited at least twice, as a result the on-feedback value is allocated by a score of 10, or a neutral score of 5 otherwise.

3. Using information of both profile and travel history The third assumption is the combination of the two above. That is, if a sight belongs to a user preferred sight group, or the user has visited at least twice without giving any score, the highest score (10) is allocated to this sight. Otherwise, a neutral score (5) is placed into a no-feedback value.

A user similarity calculation event usc_1 occurs after completing trust propagation.

ON user similarity calculation event $p_1.usc_1$
IF $\exists p \in p_1.c_1.P_1 : \exists h(p) \in H \wedge \exists h(p).loc \in S_\lambda$
DO $\forall p$ (peers of p_1): $CollectHistoricData(Fill(h(p)), H_{p_1})$, $AddCorCoef(USimilar(Fill(h(p)), Fill(h(p_1))), COR_{p_1})$

Step 3: generating collaborative recommendations collaborative recommendations are generated by predicting feedback scores for sights which the active user has not yet given their feedback score.

A rating prediction event tcr_1 occurs after completing the data updating.

ON rating prediction event $p_1.tcr_1$
IF $p_1.tcr_1.H_{p_1} \neq null : p_1.tcr_1.COR_{p_1} \neq null$
DO $CollaborFilter(H_{p_i}, COR_{p_1})$

Step 4: weight recommendations by trust The trust matrix is used to weight predicted ratings to rearrange the order of recommendations.

A trust weighting event tw_1 occurs after completing rating prediction.

ON trust weighting event $p_1.tw_1$
IF $p_1.tw_1.R_{p_1} \neq null : p_1.tw_1.T_{p_1} \neq null$
DO $TrustEnhanced(R_{p_1}, T_{p_1})$

The hypothesis of the forth approach is that correlation is existing between trust and user similarity in the personal community. With combining trust and user similarity, the accuracy and user acceptance of recommendations generated by collaborative filtering can be enhanced. In addition, the computational cost is much lower than pure collaborative filtering (pure collaborative filtering finds similar users from all users in the system, while trust-enhanced collaborative filtering looks for similar users from just a small peer group). Moreover, users know clearly about the information source and they are able to influence the recommending process through casting different personal trust or scaling the size of the peer group. Hence this solution is more transparent and controllable than pure collaborative filtering.

5 Implementation

In the beginning of this section, we briefly clarify the implementation environment of the project. Then the architecture of the trust-based recommender for TIP is described. It is followed by data modeling and data storage used in the recommendation generation. Finally, information and action flows inside the project are explained by using UML sequence and Class diagrams.

5.1 Implementation environment

This subsection briefly describes the implementation environment and structure of the TIP system.

The TIP system is implemented using the object-relational database management system PostgreSQL and Java Servlet technology. The database is the open-source software PostgreSQL with a PostGIS 0.7.5 extension for dealing with geometric data. The Java version is the java 1.4.2 standard edition. The software for running the Java Servlet is Jakarta tomcat 5.0. The user interface is a tourist web site where users can access the system using web browsers.

The TIP system has a client-server architecture. On the client side, users can access and interact with the system from the browser on either a desktop or a PDA which has been equipped with a GPS device. By using the browser on the desktop, users can easily experience a virtual travel on an electronic map, or plan their travel itineraries before starting a real journey. The approach of browsing information on a PDA is specially designed for real-world travellers on travel routes. With the wireless mobile device and TCP/IP protocol, travellers can connect to the TIP system, subscribe their request and receive results provided by the TIP server. During travels, subscribed information from the client includes users' id, geometric coordinates and demanded services from the TIP.

On the server side, the software is implemented using Java language and Apache's Jakarta Struts framework. The TIP system follows the Struts framework appearing as a Model-View-Control (MVC) architecture pattern. Within MVC framework, three modules (the *model component*, the

view component and the *control component*) are behaving independently to deal with different tasks. The *model component* is a group of Java programs created for handling business logic. It contains the core of the application's functionality. The *view component* is the interface of the system implemented as JSP programs. The view component provides the web page where users submit their requests and receive the personalized information. The *control component* receives the request from the client and then makes a decision about where to send this request. For the Struts framework, the control component is a command design pattern and is implemented as a servlet. It is configured by using a *struts-config.xml* file. This file stores the setting of the JDBC connection and the mapping of each Java Action or ActionForm bean to the JSP file.

5.2 Recommendation Architecture

Here we describe the architecture of the trust-based recommendation for TIP. Then we give a brief description about different modules inside the trust-based recommender as well as input and output information for each of them.

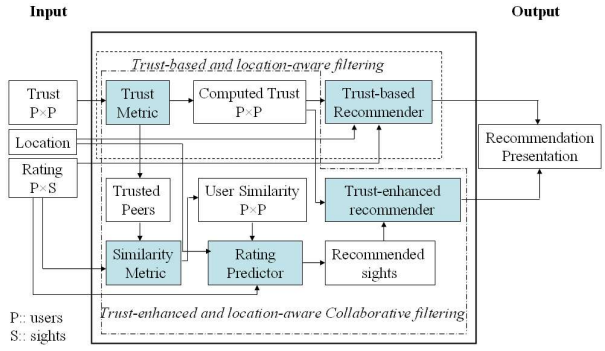

Figure 5: Trust-based Recommendation system architecture

The overall system takes input as the trust matrix (representing all trust statements in the community), the ratings matrix (representing all ratings given by users to tourist attractions) and location information in terms of users' current geographic coordinates. The system output: an ordered and categorized list of recommended sights nearby associated with information about the recommended sights and the recommenders (peers).

Figure 5 displays the architecture of the trust-based recommender system. It has two basic components (dashed boxes): the Trust-based and location-aware recommender component and the Trust-enhanced and location-aware collaborative filtering component.

Trust-based recommender component The trust-based recommender component has two modules (dark boxes): Trust metric and Trust-based recommender. The input of the Trust metric module is direct trust statements (representing by a $P \times P$ trust matrix). The trust metric module exploits trust propagation in order to predict trust for an indirectly connected peer. So it produces a computed trust matrix. The value in the cell of the matrix represents how much the system predicts that peer A may trust peer B. The trust value can be used to indicate how much the user's ratings should be taken into account when creating recommendations. The input of the Trust-based recommender is the computed trust matrix, location information about current users and historic ratings of users' peer groups. In this step, users' ratings are weighted by trust values, and then recommended sights are constructed from them. The given location is used to constrain recommended sights into a certain region. As stated in *section 4*, trust can be classified into local and global. Our project focuses on studying the local trust. We implemented three local trust propagation models.

- Propagation of Boolean trust.
- Propagation of numeric trust.
- Propagation of person-group confidence trust.

Referring to the three trust models, we developed three trust-based and location-aware filtering algorithms (TL-1, TL-2 and TL-3). Because the architecture is modular, different trust algorithms can be plugged in for different purposes.

Trust-enhanced collaborative filtering Trust-enhanced collaborative filtering has four modules: Trust metric, Similarity metric, Rating predictor and Trust-enhanced recommender. As stated before, the input of Trust metric is a set of direct trust statements and the output is computed trust matrixes and trusted peer sets. The input of the Similarity metric module is a set of peers' sight ratings. The Similarity metric computes the similarity of the current user against every other user in a peer group. It is one of the standard steps of the collaborative filtering technique. Its task is to compute the correlation between two users, then output a user similarity matrix. The input of the Rating predictor module is the user similarity matrix, the set of peers' ratings and the current location of the user. The rating prediction is the classic last step of collaborative filtering in which the rating of the active user unrated near sight is predicted. The input of the Trust-enhanced recommender is the trust matrix and a group of near sights associated with predicted ratings. In this step, the predicted ratings are weighted by trust values, then the recommended sights are organized into a hierarchy. So the output is a set of personalized hierarchical near sights for the active user. As a result, the recommendation has embedded information about user similarity and trust.

Three collaborative filtering algorithms have been implemented in the TIP system. Each of them takes different information into account to compute user similarity and predict the rating of the recommended sight:

- Recommend sights for an active user by studying users' profiles.
- Recommend sights for an active user by studying users' historic data.
- Recommend sights for an active user by studying both users' historic data and profiles.

We adopted the three collaborative filtering algorithms by introducing trust to produce three trust-enhanced and location-aware collaborative filtering algorithms (TCR1, TCR2 and TCR3) based on the second trust model.

Details on the data storage, modelling and UML activity diagrams are provided in [27].

6 Evaluation

This section firstly presents common evaluation approaches used to evaluate recommender systems. Then, we discuss difficult problems of excluding common evaluation approaches on our project. It is followed with three solutions that we propose to examine our recommender system. Finally, we give a conclusion based on the analysis of the experiment results.

6.1 Evaluation issues

This subsection briefly describes common evaluation approaches used in recommender systems followed by difficulties of performing these evaluation solutions on trust-based tourist recommender strategies.

6.1.1 Common evaluation approaches

There are many aspects of a recommender system that we can analyze. For example, the easy of use of a recommender system is one important criterion. It might require the HCI evaluation of, for example, the applied presentation and the interaction model. To evaluate the performance of recommender algorithms, techniques from Machine Learning and Information Retrieval are commonly applied, such as cross validation and measures of recall/precision. In general, these approaches are performed off-line on existing data sets. However, off-line evaluation cannot always capture users' satisfaction toward different recommendation strategies. Therefore, an on-line solution is required to evaluate on users of a running recommender system.

Off-line evaluation is to evaluate the performance of a recommendation strategy on existing data sets. In this approach, recommendation is seen as information retrieval [4], i.e., the recommendation is a subset of items relevant to the user. For this perspective, the metric is well known measures: recall and precision. An alternative evaluation approach is to view recommendation as a regression or classification problem [4]. If the recommendation problem can be regarded as the prediction of ratings likely issued by a user, it becomes a regression problem. So, accuracy of recommendation can be measured by absolute or root-mean-square (RMS) error, or by the correlation of predicted ratings with actual ratings. If the recommendation problem is seen as a classification problem. The classes are whether an item will be liked or disliked by the user. The way to perform these evaluations is similar with that used in Machine Learning: The data is separated into training and testing data. Training data is used to create prediction for testing data. On-line evaluation requires many real users to test the hypothesis, which is difficult to apply in the research community.

6.1.2 Difficulties of performing evaluation on Trust-based recommendation strategies in TIP

To evaluate the performance of a recommendation strategy, no matter which evaluation approach is applied, the first thing is to have a real running recommender system, on which ratings issued by real users to the selected items in a particular domain are able to be collected. Based on real data sets, off-line evaluation approaches can be performed to evaluate the accuracy and user satisfaction of recommendations. To evaluate the trust-based recommendation, an additional trust data set is needed to investigate the real trust relationship in the community.

The TIP system is one of the academic projects of the Information Systems and Database group in the University of Waikato and mainly used for research purposes. The first version TIP 1.0 started to be implemented at 2004. Until now it has only been developed for two years. The rating component of the TIP was finished by the end of 2005. It lets users interact with the system by expressing their subjective judgements about tourist attractions.

Because the system is still under development, we did not accumulate real ratings and the TIP community data sets. Moreover, because of domain specific requirements of the recommender algorithm, we cannot borrow data sets from different domains to do evaluation either. Consequently, the normal qualitative evaluation approaches cannot be performed on our trust-based recommendations to examine the accuracy of recommendations. Instead, three simple methods (in *subsection 6.2*) are proposed to examine our project. In the future work, with using the real data sets, three different on-line methods can be made to evaluate trust-based recommender algorithms:

- Comparison of the recommended sights generated by six trust-based algorithms against the recommendations from actual friends.
- Comparison of the recommendations generated by trust-enhanced collaborative filtering algorithms against the recommended results of pure collaborative filtering.
- Comparison of the recommendations generated by trust-based filtering algorithms against the recommendations created by trust-enhanced collaborative filtering algorithms.

6.2 Evaluation of Trust-based recommender system for TIP

For the reasons presented above, we were not able to conduct on-line accuracy evaluation. Instead we designed three simple approaches to examine the six trust-based recommender strategies: exploratory study on system functionalities, qualitative evaluation using artificial data sets and a survey for investigating behaviors of real travellers regarding taking tourist recommendations.

Sight group	Sights
Sculpture	Peace wall, Jazz Trio, Fountain-2
Painting	Civic Square 1993, When the Sunset Kissed the Mountains, Bush Walk
Mural	Waikato-1, Mural-2
Wall hanging	Evolution in our Backyard, Hamilton landscape, Balloons
Tapestry	Tubular tapestry

Table 5: Testing data

6.2.1 Exploratory study

In this subsection, an exploratory study is applied to investigate the transparency of trust-based recommendations and controllability of the recommending process.

Data setup The existing data set contains the public art attractions located in Hamilton city, New Zealand. It is provided by the University of Waikato and the Hamilton City Council. Table 5 shows a selection of the data that have been used in our testing.

Scenario setup Lan is an international student and studying in New Zealand. In the TIP on-line community, she has two direct friends, they are Hugo and Quan. Figure 6 shows the direct friend list of Lan. From the friends' information page (Figure 7), Lan can easily check where her friends went and what feedback her friends gave to the visited sights. According to the friends' travel history and given scores, Lan can issue or adjust the personal trust or the person-group confidence to each direct peer on the trust issuing web page (Figure 8).

Scenario 1: Trust-based and location-aware recommender algorithm 1 (TL-1) Having defined the peer group and issued trusts, Lan starts her tour at Hamilton with a PDA. At a paining sight *Civic Square 1993*, she connects to the TIP system. In the meantime her user id and geographical coordinates about *Civic Square 1993* are automatically subscribed to the system. She wants to request the suggestion from her friends about interesting places nearby. The trust-based recommender menu lists six trust-based recommenders. She chooses the first one (TL-1). Figure 9 displays the categorized recommended sights that are surrounding Lan's current position. Besides, the number of recommenders and computed scores for each sight group are presented on the web page. By clicking hyperlinks, Lan can easily trace detailed information about sights, sight groups, recommenders and their issued scores.

The default recommendations are created from Lan's direct friends (Hugo and Quan) only. After clicking *"More recommendations from friends of the friends"*, Lan gets more recommendations from both direct and indirect friends (Figure 10). In this time, Lan meets an unknown peer Daniel who is an indirect friend. By clicking the name of Daniel, the chain of trust path is displayed. It tells Lan that Daniel is a direct friend of Quan (Figure 10).

Scenario 2: Trust-based and location-aware recommender algorithm 2 (TL-2) When Lan checks the list of recommendations, she finds the recommended sight by Hugo is on the top of the list. Actually she prefers the recommendations of Quan to Hugo's. She wants the recommendations from Quan to be close to the top of the recommendation list. She returns back to her peers' information setting web page and issues the maximum personal trust value to Quan ($Trust(Lan, Quan) = 1$), and decreases the personal trust of Hugo ($Trust(Lan, Hugo) = 0.5$). After updating the personal trust, she chooses the recommender TL-2 to get recommendations. In this time, the recommendations from Quan are showing near the top of the recommendation list. (See figure 11)

Scenario 3: Trust-based and location-aware recommender algorithm 3 (TL-3) The recommended sights above are categorized into five sight groups (Sculpture, Mosaic, Painting, Mural and Wall hanging). Among the five sight groups, Lan is only interested in three (Painting, Mural and Sculpture) of them. However, those three sight groups are not standing close to the top of the list, and the highly recommended sight group (Wall hanging) does not meet her interests. To solve this problem, she returns back to the peer's information setting page again, then raises the person-group confidence of her direct friends on three sight groups (Painting, Mural and Sculpture), meanwhile reducing the person-group confidence on the other two sight groups (Wall hanging and Mosaic) which she is not interested in. After updating the person-group confidence values, she chooses the recommender (TL-3) to get recommendations. From the displayed results, she finds three of her favorite sight groups are close to the top of the recommended list(see Figure 12).

Scenario 4: Trust-enhanced collaborative recommender algorithm The three trust-enhanced and location-aware collaborative filtering algorithms are working in a similar manner. The difference lies in the strategy to fill the rating matrix. Our scenario illustrates the trust-enhanced and location-aware collaborative filtering (TLC-1) by filling the rating matrix using information of users' profiles.

Suppose Lan is interested in three sight categories and she has defined her preferred sight groups in her profile (Sculpture, Mural, Paining). Currently, she has visited one sight: Tubular tapestry (rating: 7). Quan has defined her preferred sight groups in her profile (Mural, Painting). And she has visited three sights: Peace wall (rating: 6), Civic Square 1993 (rating: 9), Waikato-1 (rating: 7). Hugo has defined his pre-

Figure 6: Lan's friend list

Figure 7: Quan's historic data

Figure 8: Issued trust to Quan

Figure 9: TL-1 recom.

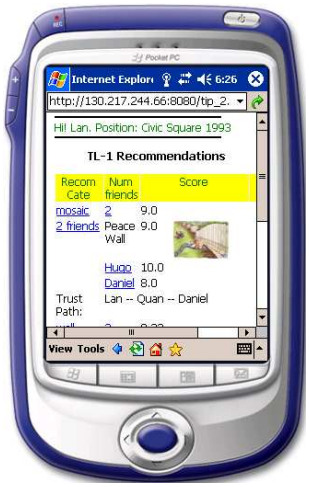

Figure 10: TL-1 recom.

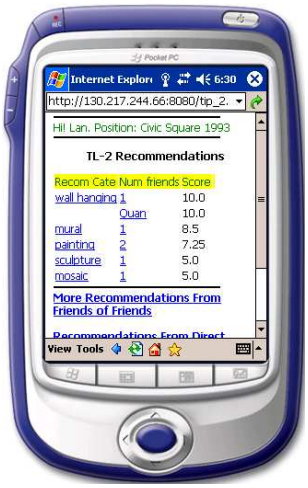

Figure 11: TL-2 recom.

Figure 12: TL-3 recom.

Figure 13: TLC-1 recom.

ferred sight groups in his profile (Sculpture, Wall hanging). And he has visited three sights: Evolution in Our Backyard (rating: 8), Tubular tapestry (rating: 6), Peace wall (rating: 8). The sight rating matrix built from peers of Lan is shown:

	Peace wall	Civic Square 1993	Waikato-1	Evolution in Our Backyard	Tubular tapestry
Quan	6	9	7	[10]	[5]
Hugo	8	[5]	[5]	8	6
Lan	[10]	[10]	[10]	[5]	7

Table 6: Scenario: Sight rating matrix

Here, scores without brackets are actual ratings issued; scores within brackets are filling scores. Computed user similarities between Lan and her friends are:
$UserSimilarity(Lan, Quan) = -0.304$
$UserSimilarity(Lan, Hugo) = -0.487$
The ordered recommended sights generated by pure collaborative filtering are:

Sight	Score	Recommender
Evolution in Our Backyard	6.9545	Hugo
Civic Square 1993	5.7727	Quan
Waikato-1	3.9545	Quan
Peace wall	3.318	Quan, Hugo

Table 7: Scenario: collaborative filtering recommendations

Trusts issued by Lan to her friends are:
$Trust(Lan, Quan) = 1$
$Trust(Lan, Hugo) = 0.5$
The resulting ordered trust-enhanced recommendation list is:

Sight	Score	Recommender
Civic Square 1993	5.1667	Quan
Waikato-1	3.8333	Quan
Evolution in Our Backyard	3.4167	Hugo
Peace wall	3.2150	Quan, Hugo

Table 8: Scenario: trust-enhanced recommendations

It can be seen that recommendations generated by both trust-based filtering and trust-enhanced collaborative filtering have provided clear reasons for recommended results. In addition, users can easily interact with the system to control the recommending procedure. As explained in Section 2, pure collaborative filtering works like a black box, it is very difficult for users to understand recommendations or control the recommending process. However, our trust-enhanced collaborative filtering uses collaborative filtering to predict sight ratings from the user's trustable peers' ratings, and then injects trusts to let recommended results be created from the most trustable and similar peers. As a result, trust-enhanced collaborative filtering is more controllable than pure collaborative filtering, and recommendations created only based on small group are easier accepted by users.

6.2.2 Quantitative Evaluation

In order to perform the quantitative experiment, we implemented several sample engines to generate data sets simulating the real ratings data and trust relationships among members in the TIP community. There are a large number of parameters existing in the system. We focused on varying the scale of the direct peer group to analyze trust-based algorithms for carrying out a recommendation in two perspectives:

- Investigation of user coverage involved in recommendation.
- Investigation of time efficiency of trust-based algorithms.

In order to get objective results, our experiment tried to cover every circumstance of the individual trust network by taking every user as input in turn to generate recommendations, with the average of results taken for comparison.

Testing environment All experiments were running on an Intel Pentium CPU 2.80GHz with 1.00GB RAM under the Windows XP operation system.

Testing data sets In testing data sets, 100 users were created for the system. Each user has a direct peer group. The maximum number of peers in each direct peer group was 3, 5, 7 and 10 respectively (it is the main parameter of testing). The total number of unique sights in the system was 500 and they were equally classified into 10 sight groups. The maximum number of ratings issued by every user was 10. The direct trust value issued was the random $real$ value between 0 and 1, and the rating value was the random int value between 1 and 10. To make results comparable, the same testing data were used to examine each algorithm in turn.

Experiment results and analysis

1. *Investigation of user coverage* Figure 14 relates the percentage of peer group that changed in each run of getting friends from the friends. The table below shows the detailed information: the average number of direct peers in each peer group and the corresponding percentage of users covered by the peer group in each recursion. The initial peer group was the direct peer group. And then the peer group was getting expanded to include indirect peers in the next level of the trust network. This process was recursively executed until the peer group could not grow further.

Analysis The chart illustrates that the speed of peer group expansion is closely related to the size of the direct peer group. It can be found, if the average size of peer group was 5.16, it covered 71% of users in the third run, while the coverage was 71% in the eleventh recursion if the direct peer group was 1.92.

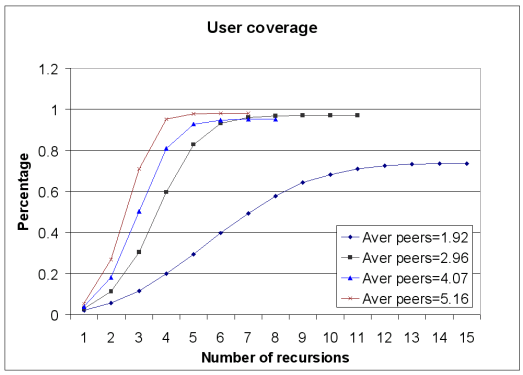

Figure 14: User coverage on different number of direct peers

This result has proven again that a small set of peers can possibly cover all users in the system if we propagate trust. Usually, users only trust a small group of peers, while a large system generated peer group is hard to be recognized by users. So regarding all users in the system are trustable peers is meaningless. To solve this problem, we have two parameters to limit the size of the peer group.

- A trust threshold is set by users to search trustable users. Since different users have different criteria to judge others, some are restricted; and some are loose. By using the trust threshold, users can define and filter out their trustable peers.
- Set the maximum number of peers in the peer group to prevent involving too many users.

According to the number of peers in the peer group, the coverage of ratings involved in the recommending process can be estimated.

2. Investigation of time efficiency of three trust-based filtering algorithms

   Three graphs (Figures 15, 16, 17) show times for generating recommendations using three different trust-based filtering algorithms in each run of recursion. In order to investigate overall performances of algorithms, we did not restrict positions of recommended sights in the experiment.

   All in all, the time spent on generating recommendations was increasing linearly along with the number of peers in the peer group. The first trust-based filtering algorithm is the most time efficient among the three, followed by the second approach, which is much faster than the third algorithm.

Analysis: For trust-based filtering algorithms, the time spent on generating recommendations is mainly depending on the trust propagation model used. The Boolean trust propagation model is used in the first trust-based filtering.

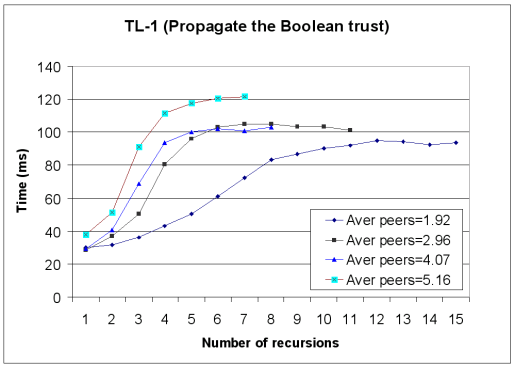

Figure 15: Comparing Trust-based recommender-1 on different numbers of direct peers

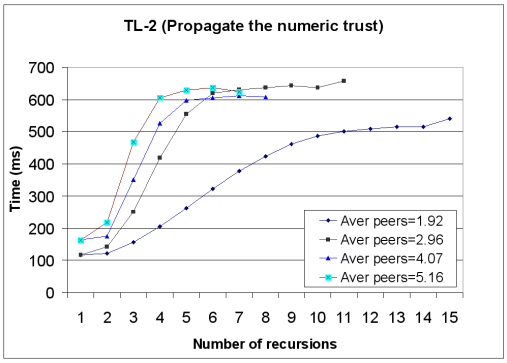

Figure 16: Comparing Trust-based recommender-2 on different numbers of direct peers

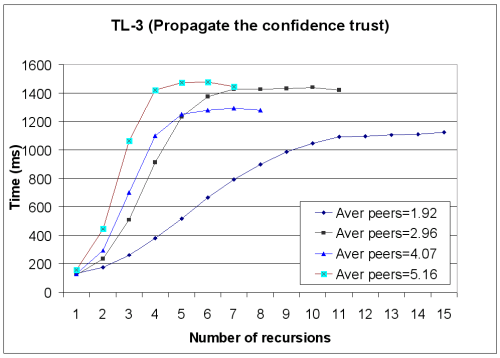

Figure 17: Comparing Trust-based recommender-3 on different numbers of direct peers

This trust model simply adds users into the peer group, which have been met in the expanding of the trust net-

work. The ratings of the peer group are treated equally in the recommendation generation process.

In the second filtering algorithm, the trust model distributes the numeric trust on the network. This model needs to predict the trust of every indirectly trusted peer. In addition, trust values are integrated into peers' ratings from which recommendations are generated. As a result, the second model involves more additional computation than the first one.

As shown in Figures 15 and 16, the second trust-based filtering takes almost 5 times as much time as the first one to produce recommendations.

The trust model used in the third trust-based filtering specifies the trust on each particular sight group. Apart from the time spent on propagating trust, more computation is needed to be spent on computing trusts on different sight groups. So the number of sight groups is another important factor which increase the time of trust propagation. Hence the computational cost of the third filtering is more expensive than the second one.

3. Investigation of time efficiency of three trust-enhanced collaborative filtering algorithms

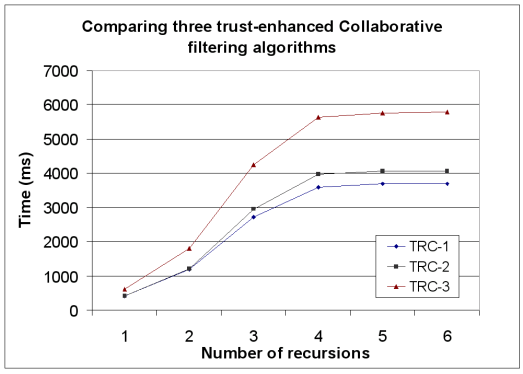

Figure 18: Comparing three trust-enhanced collaborative filtering algorithms on different numbers of direct peers

Data load: Set the number of sights (500), the number of users (100) and the number of sight groups (10) fixed. The maximum number of ratings provided by each user is 10. The maximum number of peers in each peer group is 10.

Figure 18 shows the result of comparison on three trust-enhanced collaborative filtering algorithms. Based on the same data sets, the third approach needs more time than the other two to create recommendations. While the most time efficient approach among three is the first one which is slightly higher than the second one.

Analysis: The trust propagation model used in three trust-enhanced collaborative filtering algorithms is the same. So, the time spent on trust propagation is the same for three trust-based collaborative filtering algorithms. Because the trust model provides the same users and the same number of users for collaborative algorithms in each recursion, the created sight rating matrix ($users \times sights$) is the same. Thus the cost of computing user similarities and ratings prediction are almost the same in the three collaborative algorithms.

We excluded the influence of trust propagation and the scale of the rating matrix from the computational cost. The only difference existing among the three algorithms is the method used to fill in the no-feedback value. As already illustrated in *section 4*, three collaborative filtering algorithms use three different assumptions to fill the matrix. Thus, the computational complexity relies on the information used in each assumption.

The first trust-enhanced collaborative filtering uses information in users' profiles to fill the rating matrix. The no-feedback value is replaced with the highest score (10) if the sight belongs to the user preferred sight categories, otherwise replace a neutral score (5). In this case, the time needed to be spent on checking whether the sight group has been defined in the user's profile.

The second trust-enhanced collaborative filtering uses information in the users' travel history to fill the rating matrix. The no-feedback value is replaced with the highest score (10) if the sight has been visited at least twice without gaining any score from the user, otherwise we replace with a neutral score (5). In this case, the algorithm needs to check the user's historic data, and then counts the number of times that each sight has been visited by the user. Consequently the algorithm needs to have extra time to check the travel history table and do counting.

The assumption used in the third trust-enhanced collaborative filtering is a combination of the first two. The no-feedback score is replaced with the highest score (10) if either assumption is fulfilled, otherwise it fills in a neutral score (5). So, this algorithm needs to check both the user profile and user history before replacing the on-feedback value. Obviously, the third one demands more time than the other two.

Figure 19 shows the comparison result of the six trust-based algorithms together. All experiments were applied based on the same data sets. In summary, the response time of all algorithms is growing up with the increase of the peer group scale. Among the six algorithms, the first trust-based filtering algorithm is the most time efficient. And the most computational expensive algorithm is the third trust-enhanced collaborative filtering. Overall, three trust-based filtering algorithms have much shorter response time than three trust-enhanced collaborative algorithms.

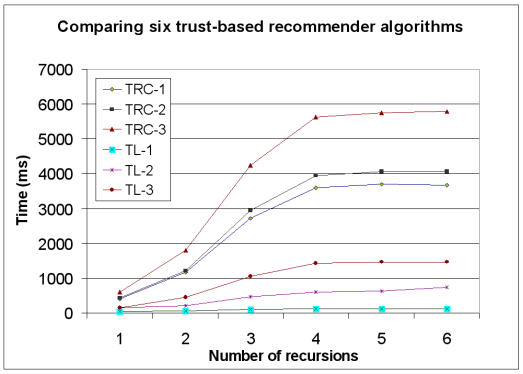

Figure 19: Comparing six trust-based algorithms

6.2.3 Survey investigation

One goal of introducing the trust concept into a recommender system is to improve the user's acceptance of recommendations. Because trust is a very human and social concept, to investigate the possibility of users' acceptance toward trust-based recommendations, we decided to conduct a survey to investigate how trust impacts people's behavior about taking recommendations in the real world, how trust propagates in the community, the attitudes of people toward recommendations from different information sources, and so on.

Based on our project, we focused on studying trust among travellers and trust-based tourist recommendations during the travelling. Participants were asked questions regarding trust and tourist recommendations from eight perspectives below:

- General information about people and their friends, especially information about people's travel habits
- Sources of tourist information people prefer to get recommendations from
- Similarities between people and their friends
- Typical factors people carefully consider before taking recommendations
- People's attitude toward recommendations from direct friends
- People's attitude toward recommendations from friends of friends
- Which conditions do people think can improve the performance of the trust-based recommender system?

In this experiment, we are interested in finding out whether the trust concept can be utilized to enhance the performance of recommender systems. Before applying this experiment, we had six hypotheses regarding trust-based recommendations below:

- People are satisfied with tourist recommendations from trusted friends rather than other kinds of information sources
- People who are in the same social group share similar travel interests.
- People only trust some rather than all aspects of their friends' lives, even though they are very close friends
- If people trust their friends, they would like to trust friends of their friends. However, people's trust attitudes to friends of friends are lower than trust attitudes to the direct trusted friends.
- Recommendations from people with high reputation are welcome
- In travelling, people would like to get recommendations from other travellers who take the same travel route.

Participants & Method: A total of 18 people completed questionnaires. Participants were undergraduate and postgraduate students from the University of Waikato. Their ages were between 20 and 40. Among 18 participants, 6 were females and 12 were males. 8 of 18 people were majoring in compute-related fields; the rest were studying in management fields. Participants were presented with a questionnaire (for details see Appendix A and [27] with seventeen questions. Participants were asked to fill in the Participant Questionnaire, and afterwards to participate in an informal conversation about further issues regarding the trust and recommendation topic that they wished to discuss.

Participants were asked about their travel history in a typical year. Thirteen of eighteen participants had taken travel for 1 to 3 times a year; four people had had trips for 4 to 6 times; one person had travelled more than 6 times in a year. To investigate trust among people, participants were asked how many close friends they had. The result was that eleven of eighteen people had 1 to 5 directly trusted friends. For five people, their close peer group sizes were between 6 and 10. The number of remaining two people's close peers was between 11 and 15. Subsequently, participants were also asked about the number of friends they knew through their close friends. For this question, most participants said that they knew more friends via their close friends. The number of indirect known friends was bigger than the number of close friends. In this way, five participants knew more than 16 friends; three participants knew 11 to 15; four people knew 6 to 10 and the remaining five people knew 1 to 5 friends.

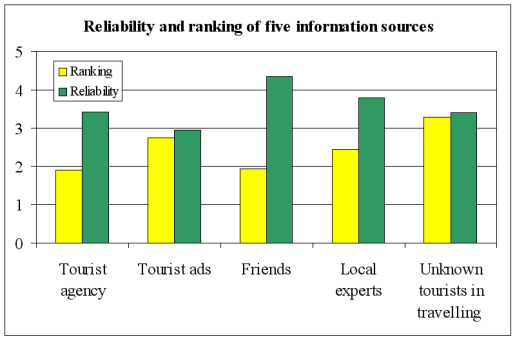

Figure 20: Reliability and ranking of five information sources

	Tourist agency	Tourism ads	Friends	Local experts	Other tourists
Ranking	1.9	2.75	1.93	2.44	3.29
Reliability	3.41	2.94	4.35	3.8	3.4

Table 9: Reliability and ranking – data

Survey 1: Reliability of tourist information sources Six kinds of tourist information sources (tourist agency, tourist advertisement, friends, local tourism experts and unknown tourists met in the same travel routes) were listed for participants. Participants were requested to give ranks of information sources that they usually consider to get tourist recommendations before travel or during travel. The rank was between 1 and 5. 1 meant this information source was the first considered by participants. And then they were asked to score the reliability of each information source. The score range was between 1 and 5. 5 meant the recommendation source was regarded as the most reliable.

Result 1: The information source of friends It can be seen from Figure 20, that a tourist agency was the first choice (it got the highest rank: 1.9) when participants required tourist recommendations. It was closely followed by the information source of friends. The average rank of information source of friends was 1.93, which was much higher than the third one: local experts (2.44). The most interesting thing was that seven of eighteen participants would like to first ask tourist recommendations from their friends ahead of other kinds of information sources.

After comparing the reliability ranking of different information sources, it was found that recommendation of friends was considered as the most reliable information and gained the biggest satisfaction. The average reliability score of friends is 4.35. Half the participants issued the highest rating (5) to it.

Analysis: It can be concluded that tourist recommendations from friends are most welcome for people. For this reason, in the tourist information system, it is essential to present tourist recommendations of friends to meet needs of travelers.

Result 2: The information source of local experts It can be seen from Figure 20 that recommendations of local tourist experts are also preferred by people. The average rank of the local experts is at the third position. Its score is 2.44 which is lower than the information source of tourist agency and friends. However, the average reliability score of this information source (3.8) is only lower than friends.

Analysis: As in the real world, the local experts in our project are travellers who have received high travel reputation in a certain area. According to this result, we can confidently say that travellers also appreciate recommendations from others who have sufficient travel knowledge of the certain area, and whose reputations have been closely examined by other travellers.

Result 3: The information source of unknown tourists in travelling The rank of recommendations from unknown travellers during traveling was quite low. People seem suspicious of information from unknown people. During the informal chatting, we were told that participants carefully considered the purpose of recommendations from unknown travellers. If they could confirm that unknown travellers were not attempting to trick them to visit some places, they were able to accept their recommendations.

Analysis: Corresponding to our project, unknown travellers on travel routes are geographically close users. It is an alternative recommendation source for travellers if they have not defined any friends in the TIP community. To increase users' acceptance, we need to present users a transparent recommendation from geographically trusted users and provide users a secure communication approach among them.

Survey 2: Investigate trust and user similarity

Part A: Participants were asked which factor (similar interests and relationship) can influence the chosen recommender, if two friends gave two totally different suggestions.

Result: It can be seen from Table 10, twelve participants favor the recommendations from friends who have similar interests with them. Four people favor recommenders who have built the strong relationship with them.

Factor	**Number of participants**
Similar behavior	12
Relationship	4
Reason of recommendations	1

Table 10: Factors of influencing the chosen recommender

Part B: Participants were asked to list three friends. And then they were requested to issue approximated trust, user similarity and trust level in a friend of each of these three friends. The range of trust, user similarity is between 1 and 5. 5 means the participant 100 percent trusts the friend, or the participant thought that she/he had the exactly same interests with the friend.

Result: It can be seen from Figure 21, the first friend of participants is the most trustable friend. The average given trust score is 4.333; the average estimated user similarity is 4.294; the average trust value of the friend of the first friend is 3.765. As a result, the first friend has gained the highest score on three perspectives. For both the second and the third friend, their trust and similarity scores are slightly lower than the first one.

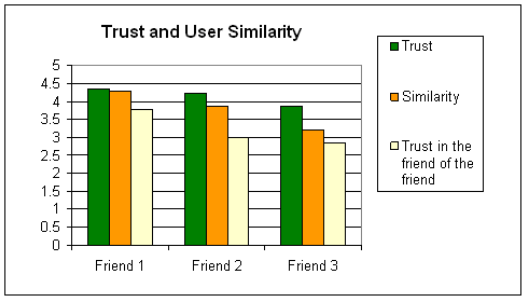

Figure 21: Trust and similarity

	Trust	Similarity	Trust in the friend of the friend
Friend 1	4.333	4.294	3.765
Friend 2	4.235	3.875	3
Friend 3	3.875	3.2	2.813

Table 11: Trust and similarity – data

Analysis: trust and user similarity According to the table above, we can observe that the most trustable friend is the most similar friend. It is proved again that trusted peers are similar to each other. This analysis result can relates to our trust-enhanced collaborative filtering.

Moreover, participants would like to trust friends of a friend. The trust value in the friend of a friend is closely related to the trust in the direct friend. In other words, if the direct friend received a high trust score, the friend of a friend also received a relatively high trust score. Trust in the friend of a friend is always lower than the directly trusted friends. Our second trust propagation model stems from this assumption.

Survey 3: Trust and recommendations of indirectly trusted friends

1. Participants were asked to give eight "Yes" or "No" answers about whether they have friends with different interests. All participants gave the same answer that they have different interests with their friends.

2. Participants were requested to give the frequency of requiring recommendations from friends with different interests. The range of frequency was between 1 and 5. The average score of frequency is 3.294, which means participants frequently request recommendations from friends with different interests.

3. Participants were asked to rate the quality of recommendations from friends with different interests. The score range was between 1 and 5. The average satisfaction toward recommendations of friends with different interests is 3.174, which is significantly higher than the average.

4. Participants were asked whether they thought about the strength of the background (e.g. experienced engineer or experienced traveller) of the friend before requesting suggestions. Seven participants always thought the strength of background of the recommender, the remaining eleven people never though about this question.

5. Participants were asked how they trusted their friends: complete trust or trust in some aspects. 14 people said they only trust some rather than all aspects in their friends.

From the set of questions and answers above, we find that people usually have some trusted friends having different interests. Because they have built strong trust relationship each other, people still are interested in their recommendations. The satisfaction toward these recommendations is relatively high. Normally, people consider the background of recommenders when they request recommendations. And most people only trust some rather all aspects of their trustable friends. According to these results, it is necessary to emphases the trust in people's preferred subjects when propagating trust and generating recommendations. This result relates our third trust propagation model and the third trust-based filtering algorithm.

Survey 4: Recommender design issues We selected several ideas about trust-based recommender system design and presented them to participants in order to get some subjective opinions from them.

1. Which type of trust value do participants think can reflect real trust using software? (Boolean or numeric)

   Participants said that classifying friends into two either trust or not trust is too simple to express their trust attitudes. In contrast, they preferred using a numeric trust value to assign different levels of trust to different friends.

2. Issue different trust values for different friends on different tourist attraction categories.

   Almost all participants agreed with this idea. They thought that assigning trust on different categories can be helpful to construct personalized recommendations.

3. Consult experts provided by the system or unknown travellers in the same trip, if users cannot get any suggestions from friends.

   For the alternative solution of no recommendations gained from friends, most participants would like to accept recommenders suggested by the system, or other travellers taking the same trip. However, they require a explanation of recommendations.

6.3 Summary of Evaluations

We performed three evaluations: an exploratory study on system functionalities, a qualitative evaluation, and a survey to examine trust in the real social community. The exploratory

study has shown that the trust-based recommender algorithms are highly transparent and controllable. The results of quantitative experiments showed that the time efficiency of the trust-based filtering algorithm depends on the trust propagation model used, while trust-enhanced collaborative filtering depends on the assumption used to replace on-feedback values in the rating matrix. Overall, the time efficiency of trust-based algorithms are dependent on the coverage of the peer group. And trust-based filtering is faster than trust-based enhanced collaborative filtering. From the survey results, we conclude people prefer recommendations from their friends over other sources. When they get no recommendation from their friends, people are willing to consult tourist experts provided by the system or geographically close travellers, but they require a clear explanation of recommended results.

7 Related work

Trust-based Recommendation in TIP is not the first recommender system utilizing trust concept. However, it is the first time to recommend tourist information by integrating users' ratings, human interpersonal trusts and spatial information. In this section, first of all we introduce a few content-sensitive electronic tour guider applications that have been implemented to provide tourist information to guide users. After that, several trust-based recommender systems applied in different domain fields are discussed. For trust-based recommender systems, we focus on analyzing trust propagation models and trust-based recommender algorithms.

7.1 Context-sensitive electronic tour guiders

A few systems for delivering tourist information have been implemented, such as the GUIDE project [15], Cyberguide project [3], HIPS (Hyper Interaction within Physical Space) project [23], Impulse project [31], Ad-me (Advertising for the mobile e-commerce user) project [14]. In this subsection, we will briefly present an overview of each of them. A conclusion is given at the end of the subsection.

7.1.1 Overview

1. *GUIDE.* The GUIDE project is a context-aware tourist guide of Lancaster City. The delivered information is "nearby attractions" surrounding the users' location. Therefore, the location information is a key factor that the recommendation generated is based on. As long as the user connects to the system using the mobile device, the location information is transmitted to the system from a wireless cell base-station network. Several base stations have been deployed through the city, and each of them behaves as a server to broadcast information about the nearby popular attractions in terms of the web page. The user is also asked to specify his profile when he/she receives the guide at the first time. After that, the system can update users' profiles by tracing users' required pages. However, this action is not automatically done by the system. The displayed information is formed in two stages: the nearby attractions are found and the order of the list is determined by filtering objects using the information of users' profiles.

2. *Cyberguide.* The Cyberguide project was originally designed for indoor usage, but it also includes an additional prototype for outdoor usage. This prototype is a mobile context-aware tour guide that delivers information based on tourist position and orientation. The position of the user is captured by a GPS unit and transformed as a latitude value and a longitude value plot on the electronic map in the system.

3. *HIPS.* The HIPS (Hyper Interaction within Physical Space) is a handheld tour-guide project designed for delivering multi-media presentations based on users' geographical positions. The software of the client side needs to be hosted on a handheld (e.g. PDA), and the associated GPS device provides the geographical location of the user. The information delivered by the HIPS server regards the physical space and the associated information. The goal of the project is to provide these two kinds of information synchronously and try to minimize the gap between them.

4. *Impulse.* The Impulse project breaks the communication into four components: the User Agent, the Wherehoo Server, the Provider Agent and the Providers. The User Agent is hosted on the handheld device: it presents the user's interests and is responsible for interacting with the Provider Agent; the Wherehoo server is a search engine running for finding information about the physical space based on the location committed as either geographical coordinates or keywords of the location; all physical resource data is stored on the Provider; the Provider Agent is responsible for interaction between the User Agent and the Provider.

5. *Ad-me.* The Ad-me project is a context-sensitive advertisement delivery system. It needs to be hosted on a PDA. The push technology is used to deliver advertisements only relative to the user's location and perceived needs. The goal of Ad-me is to intelligently select advertisements to deliver to users in a certain time, at a certain location and also to meet the users needs.

Overall, the common characteristics of these five systems are the following: they are designed for PDA or mobile device; the systems requires a thick client which means the client software needs to be hosted on the PDA. Location is the primary criterion for filtering relevant information. Four out of the five systems employ the GPS device to capture the location of the user. They are called Context-sensitive electronic tour guides.

7.1.2 Conclusion

According to our scenarios and system descriptions above, we found these tourist systems cannot fulfil our requirements on

three perspectives. First, these systems do not support a rating component. Hence, opinions or feedbacks of users about the visited places are not collected and investigated by the systems. Secondly, the displayed information follows the scheme that was pre-defined in the system or in users' profiles. Accordingly, some potentially interesting other kinds of information might not be presented to users if they do not explicitly request it, or if it is not pre-defined in their profiles. Thirdly, although with these applications users can launch self travelling, they do not have the chance to see what other travellers' behaviors were in similar trip and what their comments are.

7.2 Trust-based recommender systems

Recently, a lot of effort has been put into investigating trust in the recommendation intent to improve the performance of recommender systems, because a high trust level of interaction in the business has shown a big success. Unfortunately, most researchers of trust just remain theoretical. The only real running trust-based tourist recommender system that we can find is Moleskiing project.

In this subsection, we first analyze a real trust-based recommendation system, followed with discussions on three trust-based recommender research projects. For characteristics of trust-based recommender systems, we focus on evaluating systems on two main parts: the trust propagation model and the trust-based recommender algorithm.

7.2.1 Moleskiing

Moleskiing [25, 26] is a running Trust-aware decentralized recommender system developed at the Information and Management department in Trento, Italy. This subsection offers a rough overview about Moleskiing followed by the Moleskiing architecture, the trust propagation model and the trust-based algorithm. At the end of this subsection, we give our conclusion about this system.

Overview The goal of the Moleskiing project is to make ski mountaineering safer using the technology of information and communication. It offers an on-line community where ski mountaineers can publish and share their experiences about their skiing route. Users contribute their partial knowledge to the system about their recently experienced routes, the quality of routes and the snow condition. This information is recorded and published in users' on-line diaries. In addition, each user can explicitly express her/his level of trust on the others. Finally, users can get reliable and personalized recommendations from Moleskiing.

Moleskiing architecture Moleskiing offers a Moleskiing-hosted blog platform, the blog files are in Semantic format and automatically created for users when they come to the system the first time. Users publish their comments and ratings about their skiing routes on blog files. Each user is able to maintain the individual blog using a simple interface. Meanwhile each blog is tied with a FOAF file. The user's FOAF file keeps a list of known users associated with trust values. The value of trust indicates how much the user thinks that the other users comments and experiences are useful. Based on the lists of trustable users, a personal "web of trust" can be constructed. Moleskiing can be used in both a centralized and a decentralized environment. In the decentralized environment, users' blog files are not necessarily stored on the Moleskiing server. They can be saved anywhere on the internet. As long as users bind their blogs to the Moleskiing system, they can share their experiences and knowledge together.

Trust propagation The trust metric used in Moleskiing is called MoleTrust [20]. To the current user, the trust in an unknown peer is computed by walking the personal social network. Two steps are necessary to get the predicted trust for an unknown user. The first one is to modify the network by destroying trust cycles. In this process, a parameter of the trust propagation horizon is set for stopping the trust propagation. The default value is 3, which means the trust will not propagate when the trust distance is greater than 3, therefore a partial network is formed for the user. Subsequently, all edges from the users at level 3 to the users at level 1, 2 or 3 are removed. By now, the partial trust network becomes a directed acyclic graph. The second step is to walk on the modified network from the source peer to the target peer. The computed trust of the target peer at level x is the average of trust statements at level $x-1$ weighted by the issued trust of the peer at level $x-1$ to the target peer at level x. The minimum possible trust is 0 and the maximum is 1.

The information regarding the user's community is encoded in the FOFA file. Here, the FOFA file stores the expressed level of trust of users for other users.

Trust-based recommender The supplied recommendation comes from peers whose trusts are satisfied with a trust threshold. The detailed information regarding the trust-based recommender algorithm was not present in the paper.

Conclusion Users of Moleskiing need to manually record their skiing routes and experiences after finishing their trip. Some mistakes (e.g. typing mistakes or memory mistakes) might exist in their diaries, and some information might be lost. This prototype is not suitable for supplying an ongoing traveller. Tourist information systems associated with GPS equipment not only can reduce the input work load from users, but also precisely record users' travel routes. So precise location information of recommended sights can be presented to users.

In addition, Moleskiing was developed for providing information about skiing routes only, the displayed information is present in a plain form. The TIP system aims to offer many kinds of tourist information. It needs to classify tourism information into categories, such as history, architecture, arts etc. Thus, Moleskiing is not suitable for presenting highly branched tourist information.

7.2.2 Paradigms for Decentralized Social Filtering Exploiting Trust network Structure

This is a theoretical research project [32]. No real running recommender system is mentioned in this paper. This subsection presents a short description of the project followed by a trust propagation model and a trust-based recommender algorithm, and ends up with a conclusion.

Overview The recommendation domain of this project is books. The project aims to overcome a few drawbacks existing in common similarity-based neighborhood techniques by introducing the trust concept. It combines the Appleseed trust propagation model with a taxonomy-driven filtering technique to deal with data sparseness based on positive correlation between attitudinal similarity and interpersonal trust.

Trust propagation The Appleseed model is an adapted model based on Advogato. The Advogato maximum flow trust metric is a well known spreading local trust model used in the Semantic Web. It is proposed by Levien [18] and applied to determine the trusted members from the on-line community. This trust metric only makes Boolean decisions of trustworthiness among peers, it classifies users into either entrusted or distrusted groups. The Appleseed model integrates the numeric trust weight, trust decay and trust normalization into the trust propagation, which make rankings feasible. Appleseed operates on the partial trust graph. A predefined trust threshold is used to detect entrusted peers in the period of exploring the social network.

Trust-based recommender Ziegler and Lausen believe the feature of trust can be used to improve the recommendation quality. According to it, they described a paradigm of decentralized social filtering in paper [32]. They unify a taxonomy-driven similarity measure and the Appleseed local group trust metric into their filtering system to precisely recommend books to users. The taxonomy-driven similarity measure involves a profile generation which converts product-rating vectors into hierarchical topic-rating vectors. The topic-rating method can overcome the insufficient data overlapping problem and the hierarchical structure of taxonomy makes the similarity computation more meaningful. Person correlation measurement is used to compute the similarity of the users. The M best similar neighbors are picked as the correlated neighbors of the source peer. Finally, the recommendation is generated by considering the neighbors's proximity and the product proximity.

Conclusion Recommending sights is different to recommending books. In travelling, tourists' interests might change frequently according to the location, weather, season and so on. For example, we might like to visit historic spots in China, and do some outdoor activities in New Zealand. While interests of readers are rather steady, their preference might not change too much along with the place or the time. As a result, the recommendation domain has a big influence on the design of recommender algorithms.

7.2.3 Conceptual Framework for recommendation System Based on Distributed User Ratings

This is also a theoretical research project [16]. There is no running recommender system mentioned in this paper. This subsection offers a brief overview of the project. It is followed by a trust propagation model and a trust-based recommender algorithm called RFR. A short conclusion is given at the end of the subsection.

Overview This paper proposes a distributed recommender system in P2P environment based on FOAF. The system learns the users' preference from on FOAF-based environment where the users' tastes are asserted. The system keeps updating users' profiles in order to find a reliable user group. The method for clustering users is called cosine-based similarity. The approach of the recommender is called RFR (Recommend-Feedback-Re-recommend).

Trust propagation Because of extensibility, FOAF can be responsible for propagating trust among users. Users can weave their personal friend networks by interlinking FOAF. To improve the efficiency of recommendation in the distributed system, users are grouped before recommendation. The cosine-based similarity is used to compute the similarity between users. Finally, for user a, a set of similar users must have values of similarity greater than threshold $T_{optimal}$.

Trust-based recommender In this project, the solution of recommender is called RFR (Recommend-Feedback-Re-recommend). Three steps are needed to complete the recommending process. First of all, the recommendation, which contain three pieces of information (the name of the item, the rating result and the number of the rater), is sent from recommender user U1 (at level 0) to the users (e.g. U2, U3, U4) (at level 1), if the item rating is satisfied with a threshold. Then user U1 takes feedback about the recommended item from users on level 1. Feedbacks are ratings issued to the item by other users. Subsequently, the first recommender U1 will update the rating by averaging feedbacks. Finally, recommender U1 re-recommends the item to U2, U3, U4. Following this process, the recommendation can be transmitted to all levels. This transmision activity will not stop until the receiver has not rated the same item or the given rating is below the threshold.

Conclusion This framework provides a feasible solution to group similar users in the distributed environment and recommendations automatically transfer to the other users starting from an initial recommender. The similarity between the users and the recommender is calculated based on their preferences, and the users included in the similarity calculation are at the next level of the recommender. For example, if the level

of the recommender is 0, the users involved in the similarity calculation are at level 1 of the network.

In this project, the recommendation can only be transmitted to the undirected users through the similar users who have rated the same item and where the given ratings satisfy the threshold.

7.2.4 A Distributed Trust Model

This project is a theoretical research project [1]. Unfortunately, this paper mentions that the proposed solution has not been examined. In this subsection, first a short description of the project is presented. Then a trust propagation model and a recommendation generation procedure are presented. At the end of the subsection, there is a conclusion about the project.

Overview This project proposes a trust propagation model for a distributed environment and a protocol for recommendation. The authors point out that the common assumption about transitivity of trust (i.e. A trusts B and B trusts C, implies A trusts C) is not always true. They suggest a conditional transitivity of trust and place a set of conditions (trust categories): trust is allowed to be propagated if the trust condition has been met.

Trust propagation This project defines a trust relationship to exist between agent A and agent B, when agent A holds a belief about agent B's trustworthiness. Suppose there is a trust flow from agent A to agent D (A-B-C-D). A is interested in the reputation about D on a category, e.g. car service, A requests a recommendation from B about D on category "car service", since B cannot say anything about D, B forwards A's request to C. Because C knows about D, the reputation record about D on trust category "car service" has been stored in C's private database. Thus, the reputation about D on "car service" is passed from C to B as a recommendation and B transfers it to A; finally, A gets the recommendation which is the assessment made by C to D.

In this project, the trust of the target peer on a single trust path is calculated in the formula below:

$$tv_p = tv(R_1)/4 \times tv(R_2)/4 \times \ldots \times tv(R_n)/4 \times rtv(T) \quad (18)$$

where, $tv(R_i)$ is the recommender trust value; $rtv(T)$ is the recommended trust value of target T; $tv_p(T)$ is the trust of the target T received from the return path. If the requester can find more than one path to the target T, the average is taken as the final trust of the target T.

Trust-based recommender As described above, the recommendation is generated along with the trust flow back to the requester.

Conclusion The contribution of this approach is to provide a conditional transitive trust model in a distributed system. However, the protocol and trust calculation algorithm have not been examined according to the paper, so the performance of the trust model and the recommender are unknown.

7.3 Summary of related work

Five electronic tour guide applications supply a location-aware recommendation, which has been implemented in TIP 1.0. However, their recommendations do not consider users' opinions and feedbacks, which are the data source of the recommender service in TIP.

Four trust-based recommender projects have presented different solutions of the trust propagation model and the recommender algorithm. Because of the different applied domains, we cannot give a conclusion as to which recommender algorithm is superior to the others. Moreover, it is hard to compare the performance of the trust model.

Table 12 shows an overview of the technical criteria fulfilled in these systems and the Trust-based recommender in TIP. The "+" means that this feature is supported in the system, the "−" means it is not supported, and blank means the system does not refer to this feature.

8 Conclusion

We first summarise the contributions of this paper and then provide an outlook to future work.

1. **Six criteria considered from literature papers used to analyze recommender strategies.** We considered six criteria from literature readings that are commonly applied for analyzing the performance of recommender strategies. Then the six criteria were used to evaluate two basic recommender strategies: collaborative filtering and content-based filtering. We learned from the analysis that the two algorithms have some weaknesses for which both algorithms are difficult to overcome according to their working principles. For collaborative filtering, weaknesses of the algorithm include the fact that the recommending procedure is attackable by malicious users, the recommending process is difficult to control, there is a lack of sufficient explanation for recommended results, there is a data sparseness problem in user similarity computing, the algorithm can only be applied in the centralized system environment, etc. For content-based filtering, the main weaknesses are that some meaningful features of items are difficult to be extracted, the classification of items might involve subjective opinions, the recommending strategy has been internally defined by the algorithm, and it is difficult for users to control the recommended results.

2. **Research on the trust concept** We performed a literature study regarding the trust topic from the social and psychological perspective in the real world. The research showed that people are naturally grouped by trust, interpersonal trusted people are usually similar people, a high level of trust in exchanges can benefit both parties and reduce the cost of transaction, trust has the ability to be propagated from one to another, etc. Because of these flexibilities, some application systems

Criterion	ETG	Moleskiing	RW 1	RW 2	RW 3	TRT
Use ratings	−	+	+	+	+	+
Trust propagation model		MoleTrust	Appleseed	FOAF	Conditional transitive	TIP Trust model
Trust used in recommendation		−	+	−	+	+
System binding with GPS	−	+	+	−	−	+
Recommend tourist information	+	+	−	−	−	+
Hierarchical recommendation	−	−		−	−	+
Running system	+	+	−	−	−	+

Table 12: Recommender system overview about offered features, ETG: Electronic tour guiders RW 1: Paradigms for Decentralized Social Filtering Exploiting Trust network Structure RW 2: Conceptual Framework for recommendation System Based on Distributed User Ratings RW 3: A Distributed Trust Model TRT: Trust-based recommender for TIP

have employed trust to achieve a kind of social control over the system, such as communication software, weblogs, E-commerce applications and P2P. Those results motivate us to study trust in relation to recommendation with the aim of improving the acceptance and accuracy of recommending tourist information.

3. **Theoretical analysis of the trust-based recommender using the six criteria** The result of comparison of the trust-based recommender against the two basic recommender strategies has shown that the trust-based recommender system has advantages in a number of aspects: invulnerability to malicious attacks, the recommendation process is controllable, by using trust models users can be presented with an explanation about recommended items. However, the trust-based recommender also faces the problem of data sparseness (trusted peers have not rated any items), which the algorithm is has difficulty overcoming.

4. **Defined trust in the TIP recommender system** Apart from trust and reputation commonly mentioned in research papers, we propose two kinds of trust specially designed for the tourist information domain. These two domain-related trusts are location-aware reputation and geographic trust. By using the two trusts, users can have alternative solutions to situations where they have not yet defined their friends in the TIP community or they do not agree with recommendations of their friends.

5. **Created two trust propagation models** We created two trust propagation models: propagating the numeric trust and propagating the confidence trust. In order to let the trust model be comparable, we also implemented a well known trust model: propagating Boolean trust (Levien's Advogato local trust metric). In out trust metric, the selected trust path is the path that can propagate the maximum trust flow from the source peer to the target peer. Based on this principle, the first trust model propagates numeric trust considering the trust distance between peers. The second trust model propagates different trusts on different sight groups as well as the computed trust flow in the first model.

6. **Implemented trust-based recommenders in the TIP system** Consulting three trust models and the tourist information domain, we created three trust-based and location-aware filtering algorithms to recommend tourist attractions to users. Relating to three existing collaborative filtering algorithms in TIP, we implemented three trust-enhanced and location-aware collaborative filtering algorithms based on the second trust model.

7. **Evaluated trust-based recommenders in the TIP system** To evaluate our trust models and recommender algorithms, we applied three evaluation approaches: evaluate functionalities of the trust-based system, evaluate the time efficiency of each recommender algorithm for carrying out a recommendation, distribute a survey to collect subjective opinions of real travellers and examine the design concept of the trust-based recommender.

   Following the test scenarios, we have seen that features of the trust-based recommender have been fulfilled. From the experiment results of trust-based algorithms applying on artificial data sets, we have seen that the time efficiency of trust-based filtering is dependent on the trust propagation model used, while the performance of trust-enhanced collaborative filtering is mainly depending on the assumption used to fill the rating matrix. The survey result has proven that real travellers prefer recommendations from their trustable friends, people whose travel habits are similar. And most concepts of the trust-based system design have been recognized by participants.

In summary, we have created two trust propagation models, and implemented six trust-based recommender algorithms. According to the experiment results, we can confidently conclude that the trust-based recommender is a feasible and practical approach for recommender systems, which is superior to the other two basic recommender strategies in a number of ways, especially in tourist information domain. By studying the trust concept and requirements of real travellers, it is possible to move the trust-based tourist recommender from the centralized environment to the decentralized environment.

Future work There are a number of tasks that were not addressed so far: First, more experiments on trust-based recommender algorithms needs to be performed intensively using real world data in order to examine the accuracy of the recommendation. And HCI evaluation needs to be applied with real users, real travelling data and feedbacks from users in order to examine user acceptance.

We made an assumption that every user is able to self-publish any information and fetch information from others. However, in the real application environment, we need to carefully think about some relevant issues, such as repeated information provided to the system, secure communication using a public key to encrypt the transmitted information, policies of fetching data, etc.

The presentation of trust propagation is closely related to users' acceptance. We believe that a visualized presentation of trust propagation can be helpful for displaying trusted recommenders and assisting to explain the trust-based recommendations.

One focus of this work was local trust. In future work, we can compare the local trust metric against the global trust metric (reputation), and weigh trust-based recommendations constructed based on the local trust against recommendations created on reputation.

Another area for further research is the trust-based tourist recommender system in a distributed system environment. In real traveling, travellers may require a distributed recommender system to contact surrounding travellers in order to exchange many kinds of tourist information (e.g. pictures, video and audio). For the trust-based recommender system in the P2P environment, there are a number of challenges existing. The first one is the user privacy problem. It is a hot topic and has been intensively studied in the decentralized system and on-line community. For this problem, authentication is very important. The second challenge may be user behavior control. In a decentralized environment, every entity on the network is free to behave in any way, which might cause possible attacks on others. So a decentralized trust-based system needs to have the ability to resist possible attacks. The third problem may involve information syntax, information storage, information organizing, data caching and data forwarding between entities.

References

[1] A. Abdul-Rahman and S. Hailes. A distributed trust model. In *New Security Paradigms Workshop*, pages 48 – 60. ACM Press, 1998.

[2] A. Abdul-Rahman and S. Hailes. Supporting trust in virtual communities. In *Proceedings of the 33rd Hawaii International Conference on System Sciences*, 2000.

[3] G. Abowd, C. Atkeson, J. Hong, S. Long, R. Kooper, and M. Pinkerton. Cy-berguide: A mobile context-aware tour guide. In *Baltzer/ACM Wireless Networks*, 1997.

[4] C.Hayes, P.Massa, P.Avesani, and P. Cunninggham. An on-line evaluation framework for recommender systems. In *In Workshop on Personalization and Recommendation in E-Commerce*. Springler, 2002.

[5] S. S. Epp. *Discrete Mathematics With Applications*. Brooks/Cole Publishing Company, 1996.

[6] D. Gambetta. Can we trust trust? In *Trust: Making and Breaking Cooperative Relations*, pages 213–237. Basil Blackwell, 1990.

[7] A. Hinze and A.Voisard. Location- and time-based information delivery. In *Proceedings of the 8th Symposium on spatio-temporal databases (SSTD'2003)*, 2003.

[8] A. Hinze and G. Buchanan. Coopeerating service in a mobile tourist information system. In *Proceedings of the 13th International Conference on Cooperative Information Systems (CoopIS 2005)*. Cyprus, 2005.

[9] A. Hinze and G. Buchanan. The challenge of creating cooperating mobile services: experiences and lessons learned. In *Australasian Computer Science Conference*, pages 207–215, 2006.

[10] A. Hinze and S. Junmanee. Providing recommendations in a mobile tourist information system. In *Information System Technology and its Applications*. 4th International conference ISTA'2005, 2005.

[11] A. Hinze and S. Junmanee. Advanced recommendation models for mobile tourist information. In *Federated Int. Conference On The Move to Meaningful Internet: CoopIS*, pages 643–660, 2006.

[12] A. Hinze, K. Loeffler, and A. Voisard. Contrasting object-relational and rdf modelling in a tourist information system. In *Proceedings of the AusWeb, Gold Coast, Australia*, 2004.

[13] A. Hinze and A. Voisard. Location-and time-based information delivery in tourism. In *Proceedings of Advances in Spatial Databases: 5th International Symposium*, 1997.

[14] N. Hristova and G. OHare. Ad-me: A context-sensitive advertising system within a mobile tourist guide. In *Proceedings of 12th Irish Conference on Artificial Intelligence and Cognitive Science (AICS)*, 2002.

[15] K.Cheverst, N. Davies, K. Mitchell, and A. Friday. Experiences of developing and deploying a context-aware tourist guide: The guide project. In *Proc. of MOBICOM2000*. ACM Press, 2000.

[16] H. Kim, J. J. Jung, and G. Jo. Conceptual framwork for recommendation system based on distributed user ratings. In *Grid and Cooperative Computing: Second International Workshop*, pages 115 – 122. Springer-Verlag GmbH, 2003.

[17] R. Levien and A. Aiken. Attack-resistant trust metrics for public key certification. In *The 7th on USENIX Security Symposium*, pages 229–242. USENIX Assoc., 1998.

[18] R. Levien and A. Aiken. An attack-resistant, salable name service. In *Draft submission to the Fourth International Conference on Financial Cryptography*, 2000.

[19] P. Massa. Trust-aware decentralized recommendation system phd research proposal. 2003.

[20] P. Massa and P. Avesani. Controversial users demand local trust metrics: An experimental study on epinions.com community. In *The Twentieth National Conference on Artificial Intelligence*. AAAI Press, 2005.

[21] P. Massa and B. Bhattacharjee. Using trust in recommender systems: An experimental analysis. In *Trust Management: Second International*, pages 221 – 235. Springer-Verlag GmbH, 2004.

[22] J. O'Donovan and B. Smyth. Trust in recommender system. In *International Conference on Intelligent User Interfaces*, pages 167 – 174. ACM Press, 2004.

[23] OGrady, M.J., R. ORafferty, and O. G.M.P. A tourist-centric mechanism for interacting with the environment. In *Proc. of MANSE99*, 1999.

[24] T. Olsson. Decentralized social filtering based on trust. In *Recommender Systems*, page 83. AAAI Press, 1998.

[25] P. M. P. Avesani and R. Tiella. Moleskiing: a trust-aware decentralized recommender system. 2002.

[26] P. M. P. Avesani and R. Tiella. Atrust-enhanced recommender system application: Moleskiing. In *In ACM Symposium on Applied Computing*. ACM Press, 2004.

[27] Q. Qiu. Trust-based Recommendations in a Mobile Tourist Information System. Master's thesis, Dep. of Comp. Sc., University of Waikato, Mar. 2006.

[28] J. Riegelsberger, M. Sasse, and J. D. McCarthy. The mechnis of trust: A framework for research and design. In *International Journal of Human-Computer Studies*, pages 381–422. Elsevier Ltd, 2005.

[29] C. Shirky. What is p2p ... and what isn't? In *OReilly Peer to Peer and Web Service Conference*, 2001. online at `http://www.openp2p.com/pub/a/p2p/2000/11/24/shirky1-whatisp2p.html`.

[30] R. Sinha and K. Swearingen. Comparing recommendation made by online systems and friends. In *DELOS-NSF Workshop on Personalization and Recommender System in digital Libraries*, 2004.

[31] J. Youll, J. Morris, R. Krikorian, and P. Maes. Impulse: Location-based agent assistance. In *Proc. of the Fourth International Conference on Autonomous Agents*, 2000.

[32] C. Ziegler and G. Lausen. Paradigms for decentralized social filtering exploiting trust network structure. In *On the Move to Meaningful Internet Systems 2004*, page 840. Springer-Verlag GmbH, 2004.

Appendix A

Participant questionnaires

1. **Gender**
   Female ( ) Male ( )

2. **How many times did you go travelling in a typical year?**
   None ( )
   1−3 times ( )
   4−6 times ( )
   Other: _____

3. **Give a rough number of your close friends.**
   None ( )
   1−5 ( )
   6−10 ( )
   10−15 ( )
   Other: ( )

4. **Give a rough number of friends who you know through your friends.**
   None ( )
   1−5 ( )
   6−10 ( )
   10−15 ( )
   Other: ( )

5. **Have you requested information from the friends of the friends?**
   Never Always
   1 2 3 4 5

6. **Before the travelling or during the travel, what source of information do you usually consider to get recommendations about places you are interested in? (please give the order if you choose more than one)**

	Ranking
Tourist agency	( )
Tourist advertisement	( )
Your friends	( )
Local tourism experts	( )
Unknown tourists met in the same travel routes	( )
Other resource: _____	( )

7. **Please state your opinion on how reliable the following sources of recommendation are.**

Little reliable				Highly reliable
Tourist agency	1	2	3	4	5
Tourist advertisement	1	2	3	4	5
Your friends	1	2	3	4	5
Local tourism experts	1	2	3	4	5
Unknown tourists met in the same travel routes	1	2	3	4	5

8. **If two of your friends give you two different suggestions, which factor can influence the chosen recommender?**
   Similar interests ( )
   Relationship ( )
   Other:_____ ( )

9. **Do you have any friends whose interests are different with yours?**
   Yes ( ) No ( )

10. **Did you request suggestion from the friend whose tastes are different to yours?**

Never				Always
1	2	3	4	5

11. **How satisfied were you with their suggestions?**

Low satisfaction				High satisfaction
1	2	3	4	5

12. **Before requesting suggestions, have you thought about the strong background (e.g. experienced engineer or experienced traveller) of the friend?**

Yes ( ) No ( )

13. **If you trust your friends, do you trust them completely or for some aspects of life?**

Trust them completely ( )
Trust them only on some aspects ( )

14. **Would you like to accept the recommendation from an unknown person?**

Reject				Accept
1	2	3	4	5

15. **If the unknown person is a friend of your friend, would you like to accept his/her recommendations?**

Reject				Accept
1	2	3	4	5

16. **Please issue the trust degree to your close friends, and issue the degree of similarity between you and your friends.**

	Can not be trusted				Trustable
Friend 1	1	2	3	4	5
	Dissimilar				Similar
Similarity	1	2	3	4	5
Do you trust the	Can not be trusted				Trustable
friend of friend 1	1	2	3	4	5
	Can not be trusted				Trustable
Friend 2	1	2	3	4	5
	Dissimilar				Similar
Similarity	1	2	3	4	5
Do you trust the	Can not be trusted				Trustable
friend of friend 2	1	2	3	4	5
	Can not be trusted				Trustable
Friend 3	1	2	3	4	5
	Dissimilar				Similar
Similarity	1	2	3	4	5
Do you trust the	Can not be trusted				Trustable
friend of friend 3	1	2	3	4	5

17. **Which of the following ways would be helpful in judging the quality of the recommendation about a tourist attraction from your friends?**

Issue the Boolean trust (trust or not) to different friends

No helpful Helpful
1 2 3 4 5

Issue the numeric trust [0,1] to different friends
No helpful Helpful
1 2 3 4 5

Different trust values for different friends on the different tourist attraction category (e.g. history, architecture)
No helpful Helpful
1 2 3 4 5

Allow friends of the friends to give recommendations
No helpful Helpful
1 2 3 4 5

Consult system experts, if you can not get any suggestions from friends
No helpful Helpful
1 2 3 4 5

Allow unknown travellers in the same trip to give recommendations, if you can not get any suggestions from friends
No helpful Helpful
1 2 3 4 5