Trust-based Recommendations in a Mobile Tourist Information System

COMP594 Dissertation

Student: Quan Qiu
Supervisor: Annika Hinze

Department of Computer Science, University of Waikato
qq6@cs.waikato.ac.nz

12th December 2005

Contents

Chapter 1

Introduction

1.1 Motivation

1.2 Aims and Objects

1.3 Thesis Outline

Chapter 2

Background

2.1 Introduction

This chapter will clarify our notion of recommender system, followed with six criteria used for examining the performance of recommendation solutions. Analyze two typical recommender algorithms implemented in recommender system. Finally, introduce TIP system; and briefly describe the architecture of the system, and projects are current on going.

2.2 Recommender System

We are emerged by the data in the world especially in the internet society. Information overload problem make people feel difficulty to find the right information. There are a few applications helping people to handle this problem, such as search engine and recommender system. Search engine let the users to search information on-line, some information might not be discovered by users themselves, while the recommender system can automatically provides personalized information to a particular user.

Recommender system is an on-line application service that provides personalized and proactive items to a particular user. In general, the input of recommender system is the user's context, that will be user's personal information, the user preferred items, or the user's historic information. The output is the predicted items that the user might be interested in. Basically, recommendation takes two steps: (1) collect relevant information; (2) calculate a recommended list to the user. Normally, recommender systems are widely used in the E-commerce area. The recommended items have included movies, music, book, news web pages, etc..

Generally, two basic information filtering schemas have been used for making recommendations: Collaborative filtering and Content-based filter-

ing.

2.3 Six measurements for analyzing recommender algorithms

Before looking at the recommending strategies of Collaborative filtering and Content-based filtering in detail, we need to introduce the six criteria used to analyze characteristics of both algorithms. There are recommendation transparency, new-user problem, computational complexity, user control, malicious attack resistant and data sparseness.

Recommendation transparency
Recommendation transparency is characterized as whether the user knows who and why the certain items were recommended. Nowadays people are overwhelmed by the information coming from the different media. Some of them are obviously containing special purposes, for example, the commercial products there are usually over glorified by promoters, in order to gain the large profit from the big sale. Normally people do not want to be bothered too much by information associating with unclear intentions. Thus the recommendation, having clear source and reason for the recommended item, is directly related to the user's acceptance.

New-user problem
New-user problem refers to the issue of offering recommendations to a new user who might not have rated any product yet. As discussed above, the input of the recommender system is the information from the user, if it is the first time that the user comes to the system, and the user has not supplied any information about preferred items, certainly the system has not stored any historic information about the user. In this circumstance, a friendly system should still offer some recommendations to the user, which will be great helpful to guide the new user overcoming a cold start problem.

Computational complexity
Computational complexity is the cost of computation when offering recommendations to all users. Computational complexity is tightly related to the system response time. Having low computational load and short responding time is constantly required by the on-line recommender system. The way to reduce the computational cost is to choose an efficient algorithm, or utilize as less as possible information to accomplish computation, meanwhile the quality of the recommendation is still keeping on a certain level.

User control
User control is to see whether the user has ability to influence the recom-

mended results. This measurement is an essential problem when the user does not satisfy with generated recommendation. In addition, the performance of the recommender is able to be improved further through the user controlling, because the user provide more liked or disliked information to the system. Furthermore, if the system can provide this kind of functionality, user's confidence for approaching the system can be built.

Malicious attack resistant

Malicious attack resistant is to look at whether the recommended result can be easily manipulated by malicious users when they know the principle of the recommendation algorithm. The information security is the vital problem that users concern when they are searching information on-line. A secured system must prevent users free from malevolence. In fact, this measurement should contain the definition of the malicious behavior, however the malicious user and the fake data are extremely hard to define and detect either by the system or by the human being. But, if the system knows what data have been ruined, some algorithms can possibly eliminate those data from making decision.

Data sparseness

Data sparseness is one of major challenges for the recommender system. As users' preferences are diverse, the number of the items that the user chosen in the past only counts the small amount of the total items in the system. Subsequently, the information overlapping among the users might not happen frequently, especially in the situation where the data quantity is not large enough. If the recommender algorithm works based on the information overlapping, it might carry out none recommendation as the overlapped information can not find for a particular user. Consequently, the user, especially without computer experience, might be frustrated by none result, and eventually lead them go away from the system.

Except six measurements mentioned above, Recall and Precise are the other two criteria applied to measure the accuracy of the recommender. Recall is the percentage of relevant items that were returned, and precision gives the percentage of returned items that are relevant. Those two criteria are measurements normally using in the information retrieval. Several research papers [16, 19, 23] are also using recall and precision to measure the quality of the recommendation. In fact, recommendation accuracy needs to count the degree of personal acceptance to the recommendations, for which recall and precision might not be good measurements for recommendation accuracy. Those two criteria are excluded from our criteria.

2.4 Analyzing Content-based filtering and Collaborative filtering

Content-based filtering and Collaborative filtering are two basic strategies implemented in the recommender system, we are going to discuss the strengths and weaknesses of both algorithms based on six criteria mentioned above.

Content-base filtering
Content-base filtering studies the user's historic selections in order to suggest items similar to ones that the user liked in the past [16, 19]. To achieve it, a procedural of extracting the features of the items liked by the user is needed. For machine readable materials, such as news, or books, their features can be automatically extracted by algorithms. However, for images or movies, their features are difficult to extract from. After finishing the features discovery, we need to find out the category of the items based on their features, and then recommend other items existing in the same semantic category to the user.

1. *Recommendation transparency*
   Content-based algorithm does a good job on transparency. The user knows the recommended items are other items in the same semantic groups, which are liked by the user.

2. *New-user problem*
   This is main weak point of Content-based algorithm. The basis of Content-based schema is to detect the user's interests from the user's historic information to predict recommendations, which the user might like. If the user comes to the system at the first time, and has not selected any item yet, the Content-based algorithm is not able to provide any recommendation.

3. *Computational complexity*
   The computational cost of Content-based algorithm is heavily depending on the algorithm used to extract the features of user's preferred items. Normally, the algorithms in artificial intelligence are applied to discover the features from machine readable items. If the features of preferred items can be determined, the recommendations are items simply filtered out from the same semantic group.

4. *User control*
   From the principle of Content-based filtering, we can see that the user

can not easily control the recommender process, especially in the situation that the user comes to the system at the first time, and get none recommendation. In addition, the recommended item has been restricted by identified subjects that user preferred. For this reason, Content-based filtering might lack ability to predict new items that might potentially inspire the user to like it.

5. *Malicious attack resistant*
   Content-based filtering can not easily be attacked by the malicious user, because it only concentrates on studying each user's information individually, the recommended items are items that already have been classified manually into semantic groups, and stored in the system. So there is almost no chance for the other user influencing the recommending procedural with their fake data.

6. *Data sparseness*
   If the user's historic information is sparse (the user did not rate or only rate one or two items), it must affect the quality of the recommended result. However, the other user's data sparseness can not affect the recommendation generated for the current information demander, because this algorithm does not take the relationships among users into account.

Collaborative filtering

Collaborative filtering is also known as "social filtering" or "similarity-based filtering" [18, 23]. This algorithm is based upon a rating system in which each user is asked to give her/his explicit options of selected items in term of the numeric value. We assume if the item gained the high score from the user, the user must like it; otherwise the user will give it a low score. Accordingly the user preferred items are predicted from taste information collected from many other users.

In general, this algorithm needs to find a group of users, they are similar with the recommendation demander, and the recommendation is the collected items liked by similar users. To achieve it, the similarity calculation based on the user's context is involved. The user's context contains information of the user's preference (they might be user preferred subjects, or user's historic selection, associated with numerical ratings). Because of the suggestions gathered from the similar users, they should be liked by the user as well. In the real on-line applications, most recommender systems are implemented as Collaborative filtering. For example, Amazon.com uses collaborative filtering to recommend books based on the purchases of the

other people showing similar interest.

1. *Recommendation transparency*
   For Collaborative filtering algorithm, the process of creating recommendation is not fully transparent to the user. Although the user might have been told that the recommended items are liked by people, who are considered having similar interests with the user by the complex similarity measuring, the user has not been informed the information about the similarity, and who gave the recommendations.

2. *New-user problem*
   As discuss above, this algorithm is working based upon the existing information regarding the user. If the user can not supply any information about the individual tastes, the purely collaborative filtering is not capable to supply any personalized recommendation to the user.

3. *Computational complexity*
   Computational complexity is one of major weak points of Collaborative filtering. For Collaborative algorithm, the recommendation is generated from the similar users, so it must involve a similarity comparison. The similarity comparison should be taken place between the current user and each other user in the system. Basically, the distance metric is a common method used to measure the similarity among users. However, if the system offers recommendation to all of users, the computational complexity is $O|A|^2$ (A is the number of users in the system). It is too expensive to supply recommendation on-line. To solve this problem, Clustering algorithms in artificial intelligent are usually used to decrease the computational cost. Clustering algorithms are applied to divide all users into several natural groups, naturally grouped users have stronger resemblance to each other than the remaining users. Consequently, the information involving in computation decreases from all users to a group of users. So computational cost can get greatly decrease. Only the problem in the clustering algorithm is to decide how many groups need to be generated. Clustering algorithm is normally running in off-line situation because of low speed of calculation.

4. *User control*
   In Collaborative filtering, user control problem is related to the personal information used in the algorithm. If the algorithm utilize user pre-defined interests (the user explicitly describes preferred subjects),

the user might influence the recommendation by change her/his interests. But if the algorithm employs the user's historic data as input, the user might have difficulty to control the recommended results.

5. *Malicious attack resistant*
   Collaborative filtering can easily attacked by malicious users because of its working principle. For Collaborative algorithm, the recommended items is coming from the similar users, which means the opinions from the most similar people will be firstly recommended to the information searcher. If the malicious users know user's information, they can copy user's information to easily make a fake user having exactly same context with the current user, the fake user will be regarded as the most similar user. If malicious users make a set of fake users by copy, they can easily control the recommendation.

6. *Data sparseness*
   Data sparseness is another difficult problem that the Collaborative filtering must face, because it works based on the set of similar users. If none of other users' information can overlap with the information searcher, in other words, no similar user can be found from the system. As a result, no recommendation can be created by similarity-based algorithm for a particular user.

2.5 TIP system and architecture

Tourist Information Provider (TIP) is designed for delivering context-sensitive travel information from a variety services to travelers in their traveling route. This system is implemented as a client-server architecture, supplying both desktop computer and mobile device clients. TIP 1.0 system is the first generation and the core of TIP. It focussed on studying different event/information source, and the event-based information delivery based on the user's context (the location, personal profile describing interested (semantic) sight groups and topics, and the user's travel history), and the sight context (the location, and the predefined semantic group that the sight belongs to) [6, 9].

TIP 2.9 is extended based on TIP core (see figure 2.1), it has three layers, there are communication layer, data layer and service layer [7].

Communication layer [7] inherits event-based communication of the core TIP system, and has been extended into an event-based communication in-

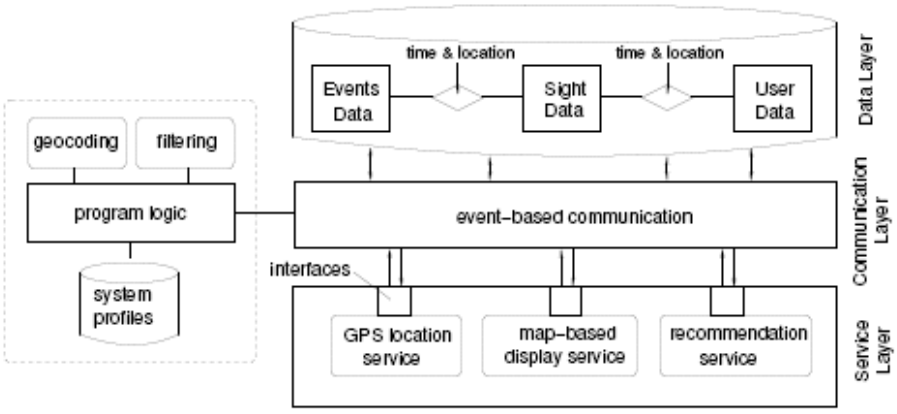

Figure 2.1: TIP 2.9 system architecture

frastructure. All events are filtered by communication layer and sent to the corresponding services.

Data layer [7] stores user-related data, the data about the sights and information regarding events.

Service layer [7] provides several different services. The communication layer is in charge of the communication among services as well as communication between the TIP core and the services. Currently, TIP 2.9 has integrated three services. There are GPS location service, map-based display service and recommendation service respectively.

GPS location service is a basic service. Through GPS device on the mobile, the geographic information about the user's current location will be recorded. Geographic information is an import parameter for communicating among the different services.

Map-based display service provides visualized information about the surrounding area in order to guide the user in the physical location. It includes labeling the user's current location, display the nearby sights as well as the sight context.

Recommendation service supplies the further interested sights to the user taking into account the user's context and the sight's context [8]. Three recommendation components has been implemented by utilizing the user's known preferences and the current context of user and sights.

There are a few other services are in the progressing. There are Trust-based recommendation, Advanced recommendation, Community-based interaction, Travel concept, Travel planning, TIP connection to Greenstone as well as Location-aware Caching in Mobile Environments.

Trust-based recommendation is the main focus of this thesis.

Advanced recommendation is taking user's profile, user's context, sight context, user's history and user's feedback into account, to find a good combination among the three recommender paradigm, content-based recommendation, collaborative filtering and knowledge-based recommendation, in order to gain effective recommender component [12].

Community-based interaction is designed for supporting a mobile virtual community for the travelers for sharing their travel experience, point of views about the sites they have been to, and voting for the comments and travelers to help the users to find out interesting destinations and whose review is trustworthy [22].

The main focus of Travel concept is to implement a travel itinerary system. A travel itinerary includes how many days this travel takes, which the places have been visited in each day and the right order which the places have been visited [21].

The Travel Planning Component aims to help tourists plan their trips dynamically on an electronic map which displays sight information associated with the TIP system [10].

TIP connection to Greenstone let the user access to a digital library (Greenstone) in order to find the information (news, maps, electronic books or paintings) relevant to the sight [5].

Location-aware Caching in Mobile Environments aims to locally store the requested information for reusing, and predict the user's movement for pre-selecting information according to the user's profile [17].

TIP is an on-line application service. On the server side, the TIP system has a database back-end using a PostgreSQL database with PostGIS extensions for the geographic data. TIP web sever application is implemented in Apache's Jakarta Struts framework. The Struts framework is a Model-View-Control (MVC) architecture pattern. MVC pattern has three separated modules: the Model component deals with business logic; the View component presents the output; the Control component is responsible for controlling flow.

On the client side, a web browser is used for displaying travel information. The client can be thin or thick depending on the services requested by the user. For example, map service requires a thick client, while TIP core system requires a thin client.

2.6 Summary

The beginning of this chapter explains the necessary of the recommender system in internet society. After that, represent the definitions of six criteria: recommendation transparency, new-user problem, computational complexity, user control, malicious attack resistant and data sparseness, utilized for evaluating the recommender solutions. Subsequently, the report of analyzing Collaborative filtering and Content-based filtering using six measurements is represented in detail. Finally, describe the architecture of the TIP system that Trust-based recommendation application bases on, and number of projects are currently in the developing.

Chapter 3

Trust problems and challenges

3.1 Introduction

This chapter will clarity the trust concept in the beginning, followed by the explanation about the reasons why the trust can solve the problems existing in the recommender system (discussed in *Chapter 2*), after that give the description about trust in TIP, as well as the problems and challenges of Trust-based recommendation for TIP.

3.2 Trust concept

Jens Riegelsberger has conducted a detail research on trust topic [11]. He discussed the trust concept from social and psychological point of view, and the positive consequence that the trust can be related, for example, trust can let the both parties better off when they are engaging in exchanges, and reduce the cost of transaction. Instead of the traditional face-to-face communication, new on-line technology becomes a new media for people's interaction. Researchers have observed that supporting trust and trustworthy technical can make the technology-mediated interaction going smoothly, E-commerce is one of examples, and it only can run successfully by providing users a high level of trust interaction.

Trust is a hot topic and has being intensively studied by researchers in the different fields. For the Trust-based recommender system, we prefer to use Diego Gambetta's definition about 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 to be able to monitor it) and in a

context in which it affects [our] own action" [4]

After analyzing six on-line recommender systems, three book recommender systems: Amazon.com, RatingZone and Sleeper, three movie recommender systems: Amazon.com, movieCritic and Reel.com, Rashmi Sinha and Kirsten Swearingen gave conclusion in their paper [20]: the recommendations provided by the user's friends are consistently better than the recommender system. The reason is that the recommender system only have limited, domain-specific knowledge about the users, however friends know the user, and have sufficient knowledge about the user's tastes in a number of domains, their suggestions can easily acknowledged by the user. But the user is also interested in the items recommended by online recommender system as they are able to offer "new" and "unexpected" items, while friends might recommend the user the items that are previously identified interests. This report presents the confident necessity of studying the trust concept in the real world, and the essentiality of improving the performance of the recommender by integrating the trust concept, in order to supply personalized and precise recommendations to users.

In the real world, trust is produced during people's exchange [11], and people like to act upon the recommendation form entrusted people (their peer groups). Before people take the suggestion of peers, three promises have already been in their minds: firstly, they trust the recommenders; secondly, they assume that the recommender has sufficient knowledge of their tastes or tastes of people like them; thirdly, they assume that the recommender has knowledge of the alternatives available.

Trust is a concept normally used in human society. We are not going to deeply study the trust concept from physiologic and social point of views. We are interested in what common things sharing among the entrusted people, what peers' behaviors can inference each other, how to automatically describe this kind of relationship with software, how to take the trust information into account to benefit all users using their experiences and assessments, as most of them might never meet before in the internet community, and how to let the user meet as many as possible potentially entrusted and similar people, and prevent the user free from the attack by intrusted people.

Although everyone has particular definition of the trust in their mind, naturally grouped people by the trust always hold some positive attitudinal similarity [14]. For example, some people fully trust each other, it is because they are coming form the same family, or they have similar interests, say come from the same club, or they have similar background, say being in the same age, or studying in the same school, or serving for the same company. Because of sharing similar judgement on some aspects among peer group,

their behaviors or interests can influence each other without doubt. Once trustworthiness has been well constructed between people, the suggestions are easily exchanged by trusted people.

In addition, information reliability is carefully concerned by people when they are look for the short cut to the problem, on which they might lack enough knowledge. Although there are various sources of information available, such as internet, television or newspaper, people still prefer to the direct suggestions from someone they know well. For example, when we decide to diet out, in general, we firstly ask for the suggestion from the people around us, and act based on their recommendation.

The trust can propagate through peers to peers, by using it, an individual trust network will be weaved for every single user. For example, A trusts B, and B trusts C, it might A trust C as well, even they do not know each other. In fact, people develop their social networks in the same way, in order to meet new peers, and get more information.

Nowadays, the concept of trust has been introduced and studied by researchers in computer science, and several application systems have integrated the trust concept into the system. For example, online communication software, MSN (messenger.msn.com), it allows users to contact their friends around world at anytime. The trust can be seen as the users permit the other users to connect them, and block the contact from intrusted users. MSN helps the user to build a personal community, while Hi5(www.hi5.com) helps to extend the independent personal community to a social communication network. By using it, each user can meet new unknown friends through their peers. Weblogs (news.bokee.com) is another example, the number of clicks and the number of reviews of the users on each topic is recorded by the system. Accordingly list the top 15 hot topics in the last 48 hours or the last week on the web page. In this case, the trust is acted as reputation in the system [14], which is used to guide readers. Similarly in the E-commerce area, like Amazon.com, the reputation of the product is computed by the number of starts given by the custom or the reader. Obviously, the number of stars gives great help to the person, who wants to find something, but lacks sufficient knowledge about it on the internet society where the information is overflowed. For the decentralized architectures, such as Peer-to-Peer application, the trust concept is used to distinguish the trustable equipments on the internet to avoid the risk of virus or other attack.

Currently most on-line recommender systems use reputation. Reputation is a concept that is similar with trust, but they have slightly different meaning. Trust involves two peers, each peer hold a particular trust value about the other, which means each user will gain the different trust values

from the different users; while reputation is a property of the peer assigned by the embedded social network [14], it means each user in the system will be granted a global value of reputation by the system.

3.3 Trust can solve the problems

We are going to use the same criteria defined in *Chapter 2* to prove that only trust enhanced recommender can solve the problems existing in the Collaborative filtering and the Content-based filtering algorithms.

Recommendation Transparency
Trust-based algorithm provides sufficient transparency to the users, while lacking transparency is one of main weak points for Collaborative filtering. Normally, Collaborative filtering is operated like a black box, only present the final results to the users. Being short of solid reasons for the recommended items might be hard to convince the user to follow. Nevertheless, Trust-based recommendation gives clear cause and the recommenders to the user. The user knows who gives recommendation and what that is. It is easy for the user to make a decision whether the user need to follow. If the user has question about the recommended item, the user can check the recommender of the suggested item from the trust-based system. If the user has problem with the recommender, the user also can track back to find the trust path that lead the user reach to an unknown recommender.

New-user problem
The risk of the new-user problem is much smaller in Trust-based algorithm, however it is the major weakness of the Collaborative filtering and Content-based filtering. For this problem, both Collaborative filtering and Content-based filtering are difficult to conquer. Theatrically, if the user has at least one friend in the peer group, this problem can not happen in the Trust-based algorithm. In split of this, if the new user does not define any friend yet, the personal peer group will not be able to build up for the user. Consequently, it is impossible for the user gaining any recommendations from her/his peers. Similarly, if the user has totally opposite tastes with peers, the user might not obtain the satisfied recommendation as well. In those two typical cases, the personal trust might not work well. To solve this difficulty, we can use reputational trust or domain-related trust instead. By using those two kinds of trust, the user also can get recommendations from the users who have been strongly recognized by other users. As a result, the quality of the recommendations can still keep on a high level.

Computational complexity

The computational cost of the Trust-based algorithm exists in the different aspect with the similarity-based algorithm. For the similarity-based recommender, the computational complexity critically depends on the quantity of the information (the total number of the users in the system and the number of the recommended items) and the algorithm used. There are a few solutions used to look for the similar users, such as algorithms in artificial intelligent, but they only can run off-line situation because of low speed of computing. However, the computation scale for generating recommendations in Trust-based algorithm is smaller, because the mount of data involving in recommendation generation is completely depend on the size of the peer group, although the large size of the peer group causes more intensive computing, the coverage of the peer group is hardly to reach 1. In addition, the size of the peer group can be easily controlled either by the user or the system, it is possible to find a trade-off way to balance the computational intensive and the quality of the recommended results. Further more, the computational complexity of the Trust-based algorithm also needs to include the trust generation for each individual user. Because the system needs to generate a set of trusts for each user individually.

User control

As discussed in *Chapter 2*, the algorithms of collaborative and content-based filtering acted as a block box. When the recommended items are getting bad, it is difficult for the user to interact with the system in order to influence the results. However, Trust-based algorithm presents the user a transparent procedural, the user can simply interact with the system through a few ways, such as expanding or narrowing down the size of the peer group, issuing explicit trust to each peer, or trying to influence the order of recommended items in different types of trust.

Malicious attack resistant

Defining and detecting malicious behaviors is hard job even in the real world. If the data has been ruined by malicious users, those fake data might have big inference on the recommendations generated by similarity-based algorithm. Using the Trust-based algorithm, this problem can be easily prevented by excluding the fake users from the peer group or decreasing the trust on them, if the malicious behaviors have been detected by the user.

Data sparseness

Data sparseness will cause problem on the quality of the recommendation using Trust-based algorithm. If user's information does not overlap with her/his peers', peer's suggestion might be disliked by the user, but the user might tolerate this result as she/he knows her/his peers are interested in the different areas. Sometimes, their suggestion might inspire the user to take recommendation. It is because they have constructed firm trust relationship.

	RT	NUP	CC	UC	MAR	DS
Content Filtering	+	-	+(-)	-	++	+
Collaborative Filtering	-	-	-	-	-	-
Trust	++	+	+	++	++	+(-)

Table 3.1: The comparison among three approaches on five aspects (RT: Recommendation transparency; NUP: New-user problem; CC: Computational Complexity; UC: User control; MAR: Malicious attack resistant; DS: Data sparseness
Symbols: "-" means having disadvantage; "+" means having advantage; "++" means performance is better than "+"; +(-) means either disadvantage or advantage can appear).

Table 3.3 gives the summary of the properties regarding three algorithms. It can be seen that the performance of trust-based scheme is superior to pure Content-based or pure Collaborative filtering on almost all aspects. This comparison result is the initial motivation to utilize trust concept to enhance the recommender in TIP system. Our goal is not only generate personalized recommendation to meet the needs of each individual user, but also increase user's acceptance of the results in order to boosting the confidence of the user to the system.

3.4 Trust in TIP

The goal of the recommender system is trying to give valuable and acceptable recommendations to the information demander in an efficient way. However, the recommender application is limited by recommended items, different kinds of recommended items need different requirements. The recommender application in TIP is to recommend sights to the travelers, so the requirement of recommending sights should be different from recommending books, CD or news. In order to build a trust model for giving recommendation regarding sights, three kinds of trust are worthy to study. There are personal trust, reputational trust and geographic trust.

Personal trust
Personal trust is local trust statement, that is personalized and subjective view depending on the evaluation of the individual user to the other users, according to the quality of perceived information. Each user might have different trusts from different users. In the most cases, the user can only has direct judgement about small mount of users. Although the remaining users are unknown to the user, the user mgiht reach them through a trust

network. We call it as the *trust propagation*. Following the trust network, a trust score is able to be predicted from the user to an unknown one. Trust-based recommender system focuses on analyzing the peers' behaviors, and then recommends items liked by the peers to the user. According to the research paper [15], the local trust (personal trust) achieves higher accuracy than a global one (reputational trust). In general, the local trust can be more precise and tailored to the single user, but the computational cost is more expansive since the system must work out the trust for every single user. This thesis focuses on studying the personal trust, trust propagation, the computation of the trust, and builds a trust model specify on suggesting travel information.

Reputational trust
Reputational trust is a global trust metric approximately computed by the community as a whole to a specific user [14]. As users' reputations are closely examined by all other users in the system. We can say the user who has high reputation should have sufficient domain knowledge on recommending items, and their advices can benefit other users. We call those people are experts. Ye has implemented a Community-based interaction for TIP [22]. This project includes a review system. The review system let each user publish individual reviews about sights, all reviews are public to all users in the system. And each user can rate the reviews of other users. Reputation of a single user is computed according to the ratings of the user's reviews given by other users.

Comparing to the personal trust, reputation is more objective. Users who have high reputation mean their contribution is recognised by other unknown users. Reputational trust gives the users an alternative choose when they want to look for the suggestion from domain experienced users.

On the other hand, introducing reputation into the system can encourage users behave well, in other words, let users be responsible for their behaviors in on-line society.

Although the credibility of reputation might not be compatible with the personal trust, the proportion of user coverage is higher than the personal trust, usually it is close to 1 [15] , and the cost of computation is much lower than the personal trust, as the system only needs run algorithm once to obtain reputations of all users.

Geographic trust
Geographic trust can be seen as domain-related trust. Obviously, different domains of recommendation possess different requirements. We might need not consider the geographic difference when we recommend books, but we

do need to carefully concern geographic information about each sight when we recommend travel information. For example, if the traveler is living in China, and currently having a trip in New Zealand, the most suggestions from Chinese friends might not be useful to the traveler. Instead, the traveler might like to get recommendations from other travelers who are currently having a trip around, or the local residents, even they are not known each other. Besides the geographic factor, other factors, such as the season, weather and opening time, are also needed take into account in the recommending process when we suggest travel information. However those factors are not counted in this thesis.

3.5 Problems and challenges of Trust-based recommendation in TIP

Before building a trust model for generating Trust-based recommendation about sights in TIP, a few problems we need to carefully think about:

1. How to decide a proper trust of the source peer to a direct trust peer? The trust is explicit rating assigned by the source peer to a target peer. If the target peer is the direct friend of the source peer, the source peer can directly issue a trust to the target peer according to the personal assessment.

2. How to predict a trust of the source peer to an indirect trust peer? Generally, the number of user's direct friends is much smaller than the total number of all users in the system. If the peers' data is relatively stable, only concentrating on studying the direct peers' data might not always get "new" recommended items. One way to solve the problem is to enlarge the user's peer group to include more potentially trustable peers. They might be unknown by the user, but they are entrusted peers, as they have been examined by user's peers.

   As the scale of the peer group getting larger, the distances of the source peer to some target peers are further than some ones. Psychologically, the trust relationship is getting looser when the distance between peers is longer. We need to create a formula to simulate this trust situation. We call it as *trust decay*. The predicted trust of the source peer to an indirect trust peer should combine the explicit trust value with the trust decay.

3. How to decide a set of confidence values of the source peer to a direct

trust peer?
There are not two identical people in the world. For this reason, people's interests are different, even between two most trust peers. The confidence value is issued by the source peer to a target peer on one specific aspect. There will be a confidence vector formed from the source peer to the target peer on all subjects. It not only describes the assessment of the source peer to the target peer on number of subjects, but also implies the interested subjects of the source peer.

Introducing the confidence into the Trust-based recommender system can create more personalized and precise recommendation to meet the user's requirement. For the direct friends, the source peer can directly assign the confidence values on different subjects about the target peer according to the perceived information.

4. How to predict a set of confidence values of the source peer to an indirect trust peer?
   For the indirect trust friends, the predicted confidence values should combine the confidence values issued by the direct friend (who is one of peers in the source peer's peer group) of the target peer with the trust decay of the source peer to the target peer.

5. What is the recommended result?
   The Trust-based recommendation contains two kinds of results. One is ordered sights list, and another is the set of recommenders. The system has invited users to express their opinions about sights in terms of numeric feedbacks. And the trust information regarding the source peer to the target peer is also the numeric value. The recommended sights should integrate the feedback and the trust to come out an ordered sight list associated with the computed score. As discussed before, Trust-based recommendation is generated from the source peer's peer group, which involves trustable and potentially trustable peers together. To make the recommending source clear, it is necessary to present the user a trust chain of the source peer to the target peer.

6. Get recommendations from the user's peer group, which includes direct and indirect peers together.

Scenario 1: Currently, Jane has one day trip in Hamilton. She wants to visit one or two spots, but she does not know anything about Hamilton. She turns to ask help from her friends on mobile tourist informa-

tion system. From the recommendations, she finds Hamilton Garden got the highest score, staying on the top of the sights list, and the recommenders include two direct friends and a few indirect friends. Because Hamilton garden is strongly suggested by friends, she decides to visit there.

7. Control the order of recommendations by the trust of the source peer to the target peer
   Psychologically, the user likes to take the suggestions from the friends they are quite close to the user. For this reason, the recommended sights by trust most peers should close to the top of the list. And the recommender for the top sight should gain high trust from the user.

8. Control the order of recommendations by the confidence of the source peer to the target peer on certain subjects.
   Basically, people want to get recommendations from most trust peers and also match their interests. In this approach, the top recommended sight should much fit the user's tastes, and the recommender should win the high trust from the user.

9. Recommendations from the users who have high reputation
   If the user's peer group can not provide useful information, or the user has not got any friend from the system, the user might like to take the suggestion from the users who have been evaluated and recognized by other users as a domain expert. In this case, a reputation system is required to generate a global reputation value for every single user.

   Scenario 3: Lan is an international student from China. She wants to go for a holiday in Hamilton. From the TIP system, she can not get any suggestion form her friends because most her friends are studying in China. But the recommender system provides her recommendations from the users who are regarded as travel experts about Hamilton.

10. Recommendations from the users who are geographically close to the traveler.
    In the real traveling, the travel route will be affected by the weather, the temperature, the opening time or other unpredicted reasons. In this case, other travelers who are in the same location might hold up to date information about sights, and most interesting places from their personal point of view. The comments and suggestions from the

travelers in the same travel route are considered as valuable recommendations for users.

11. Improve the performance of the collaborative filtering by trust.
    Collaborative filtering creates recommendations for a certain user from other similar users. The computational cost for finding similarity is a limitation of the algorithm. However, this problem can be solved by intergrading trust concept into it, since peer group shares some positive attitudinal similarity among them, they can be regarded as similar users. For this reason, the performance of collaborative algorithm can be improved by integrating the trust concept.

12. Influence of recommendations:

    (a) The user has similar interests with some peers in the peer group.
    Scenario 2: Lucia is specially interested in surfing. In the last holiday, she went to a famous beach recommended by a tourist agency. But she found there was so crowded own to its famous reputation. She did not have a happy time there. She knows lots of her friends have same interests. From the mobile tourist information system, she found a few friends strongly recommend another surfing place. She plans to go there in this weekend.

    (b) The user has totally opposite tastes with peers.
    This is difficult situation for Trust-based recommender and hard to solve. Although the user might not satisfy with the recommended items, the acceptance of the frustrated items from the trust algorithm is still higher than the frustrated recommendation gained from collaborative or content-based filtering. Because the user trusts recommenders.

13. Data privacy:
    The personal data privacy is essential problem for on-line system. People might allow some not all of their data to be accessed by peers, or they might not want to share their data to the other people else, even those people have been labeled as the friend of the friend. Therefore we need to carefully consider data privacy problem in the system. In this project, we assume all data are public to all of users, each user can access any data from any user.

14. How to measure the quality of the recommendations?

15. New-user problem
    If the new user has not defined any friend, the recommender system can not build a trust network for her/him. Alternatively, the new user still can get recommendations from reputational trustable user or geographic trustable user.

16. Define user's direct peer group.
    The direct peer group is fundamental factor for building a trust network for a certain user. Four ways can help to pick the friends from the system. Firstly, the user can directly pick friends from the system, if the user knows some ones. Secondly, the user can pick unknown friends in the trust propagation based on their recommendations. Thirdly, the user can choose other users who have high reputation as friends. Finally, the geographically close user is one of entrusted friend sources, the user also can pick the trustable friends from geographically closing travelers in the traveling.

17. Trust value updating
    The trust value should be automatically updated according to the user's opinion about the recommended sights. After the travelers visited the recommended sight, they will be required to provide a feedback in terms of numeric value about recommended sights. We can evaluate the user's satisfaction based on the feedback score about the sight. For example, if the feedback score is above 7 (the feedback range is from 0 to 10), we say the user satisfy the recommendation, the system can automatically raise the trust value of the peer who gave the recommended sight, meanwhile increase the confidence value of the source peer to the target peer on the corresponding sight subject. If the feedback is below 2 or equal to 0, we say the user does not like the recommended items or she/he is not interested in this subject, system will automatically take some trust or confidence credit off from the recommenders.

    User's satisfaction is one of main measurement [2] for the Trust-based recommendations. The definition of the satisfied or dissatisfied behaviors will not be discussed in this thesis. Automatically updating the trust and confidence values can be helpful to detect the user's interested subjects, also find out the similar users or the user preferred recommenders.

3.6 Summary

This chapter presents the trust concept, and this concept has been applied in the on-line application systems. Five criteria of analyzing the collaborative filtering and content-based filtering are used to examine the Trust-based recommender, and confidently point out that trust can have better performance than the other two. For the Trust-based recommender in TIP, three kinds of trusts utilized to generate Trust-based recommendation. And the challenges that Trust-based recommender will face.

Chapter 4

Design of Trust-based Recommendation for TIP

4.1 Introduction

This chapter will introduce a few terms and definitions mentioned in this thesis for representing the trust concept, after that four approaches of Trust-based recommendation for TIP system will be explained in detail.

4.2 Terms and Definitions

This section describes the precise terms and definitions of the key Trust-based concept used in this thesis.

Sight
Sight set $S = \{s_1, s_2, \ldots, s_n\}$, contains all sights. Each sight $s_m \in S$, is a place where the traveler might like to visit.

Sight group
Sight group set $G = \{g_1, g_2, \ldots, g_k\}$, contains all classifications of sights. Each sight group $g_l \in G$ represents one specific category that sights $s_m \in S$ belong to. Sight groups can be broad or narrow categories. We set θ is the direct set of sub sight groups, $\theta(g_l) \subseteq G$, $g_l \in G$. And all direct sub sight groups, $g_l, g_k \in G$, $l \neq k$, have $\theta(g_l) \cap \theta(g_k) \geq \emptyset$. All sights in one sight group are sharing the similar topic. We call the sight group as the semantic sight group.

Nearby sight group
Nearby sight group S_λ, $S_\lambda \subseteq S$, contains sights geographically closing to a given location. λ is the distance threshold utilized for computing the nearby

sights.

Feedback
$F = \{f_1, f_2, \ldots, f_n\}$. Set F includes all feedbacks of all travelers to the visited sights. The feedback f_m, $f_m \in F$, is an individual opinion issued by a traveler to a certain sight. Each feedback statement is represented as a numeric value ranging from 0 and 10, it is applied to reflect the assessment of the traveler to a sight. For this reason, set F is considered as a set of judgements or knowledge to sights. Those judgements can be valuable, they might help other travelers making decisions in their traveling.

There is an assumption about the feedback in the project, we assume that every traveler issues the feedbacks to the sights, all feedbacks are public to all travelers, and each traveler can feel free to access and fetch any data from others'. Here we do not consider the problem of data privacy.

Peer
Peer p is the globally unique and independent identity of the system. One peer represents one user of the system.

Peers
$P = \{p_1, p_2, \ldots, p_n\}$, set P contains all peers. There are in total n (finite number) peers in the TIP system.

Peer group
Peer group is a specific social community of the peer. People who are in the same group are called direct or indirect friends (A definition of direct/indirect friends is given below) of the peer. Each peer has own peer group that is a sub-set of P. Suppose there is a peer, p_i, $p_i \in P$, the corresponding peer group is referred to as P_i, and $P_i \subseteq P$.

Personal Trust
In the real world, if someone is well known by the other people, they must hold the different 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 human community. Personal trust is the individual judgment of one to the other. We define that the personal trust has five properties below.

1. Typically, peers only issue a personal trust score to a peer that is the direct acquaintance. It is to mimic the real assessments between people regarding interpersonal relationship in the real world, because people only can directly express their opinions about their direct friends. Consequently trust value is heavily depending on the individual's opinion

and evaluation, thus it is subjective.

2. Personal trust T_p is a real value between 0 and 1, which is used to explicitly illustrate the individual trust relationship between two peers. For the whole peer set P, the personal trust T_p can be formulated as:

$$T_p : P \times P \rightarrow [0, 1]. \tag{4.1}$$

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 , it might because the source peer knows they hold the similar hobbies or tastes. Consequently, the suggestion of the target peer might be firstly taken by the source peer.

$T_p(p_a, p_b) = 0$ implies two possible reasons, the first one is that the source p_a might trust the target peer p_b very well, but they possess totally opposite interests; the second one is that the source considers the target peer as a malicious user, and this target peer will be excluded from the peer group later on. In those two cases, the source peer issues the lowest trust to the target peer, intents to indicate that all recommendations from this peer will not be accepted by the source peer. The peers outside the peer group always hold the trust value is *null*.

Different to our approach, Levien's Advogato [13] trust metric only makes *boolean* decisions regarding trustworthiness, it directly classifies the local groups into entrusted or intrusted ones.

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

$$p_j \in P_i \quad iff \quad T_p(p_i, p_j) > 0; \quad i, j \in [1, n], p_i, p_j \in P. \tag{4.2}$$

4. The personal trust is not symmetrically distributed between two peers. For example, the source p_a trusts the target peer p_b in a certain level, which does not mean that p_b trust p_a in 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 is different from the similarity used in the collaborative algorithm. Similarity is symmetrically distributed, and calculated by using various strategies. Two different users share one value of the similarity.

5. Personal trust is a customized and subjective value, which is reflection of the individual evaluation and assessment. For this reason, it should be defined and manipulated easily by the users themselves. By setting and operating the personal trust value, the user can interact with system to influence the recommendation process, meanwhile the user also can free from the attack by the malicious peer.

Reputational Trust

Reputational trust T_r is a global trust value for each user in the system. Currently, two projects named Community-based interaction and Travel concept for TIP([21, 22]) are in progressing. Both projects include a voting system, it let users evaluate the personal contributions (travel comments, reviews and the itineraries) each other. The reputation of the user is computed according to the judgements given by the other users. Because the reputation is closely examined by users, it is the combination of all other users' opinions to a particular user, can be regarded as an objective assessment to a single user. We assume the user who gains the highest reputation is been considering as having sufficient knowledge regarding traveling. We can call her/him as travel expert. The suggestions of the travel expert are valuable for all other users, and should be taken for recommendation propose. From the trust point of view, we call the reputation as the reputational trust corresponding to the personal trust (local trust).

$$T_r : P \rightarrow [0, 1] \tag{4.3}$$

Geographic trust

Geographic trust T_g takes the geographic information about the users into account when creating the recommendation. Recommending sights is different from recommending books, CD or movies, it has been critically restricted by domain information, such as geographic information or visited time. And the goal of recommender in TIP is to suggest sights to the travelers where they might be interested in further, based on the location where the traveler is at. Further more, in the real world, people might have a travel in every a few months, or even several years. The sight information provided four months or one year ago might not be correct any more, while the travelers, who are geographically near to the user, must hold the newest information regarding the nearby sights. As a result, we can confidently say: if the travelers are traveling around and close to each other, those travelers are seen as useful sources for providing recommendations, because they have held the newest information about sights nearby, their opinions and assessments should benefit each other; in addition, they also are able to seen as a group of similar users, because they have similar travel interests, their travel behaviors are worthy to study.

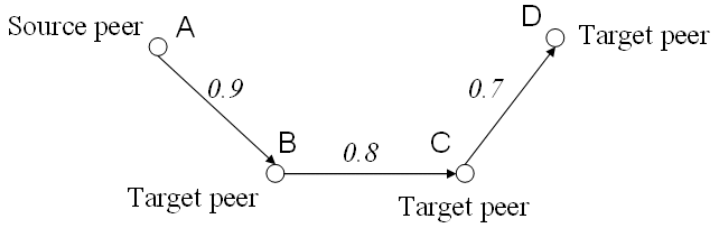

Figure 4.1: Trust path graph

$$T_g : P \times P \rightarrow [0, 1] \tag{4.4}$$

This thesis focus on studying the personal trust in detail. Because of the time limitation, we do not expand the trust to include the reputational trust and geographic trust. However, we recognize those two kind of trust are essential for offering recommendation to the tourist in their traveling route.

Trust path
A trust path graph ρ (see Figure 4.1) [3] 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 trust path can be described as the list of vertices:

$$\rho[p_0, p_n] = p_0 - p_1 - \ldots - p_n, \tag{4.5}$$

where the $p's$ represent vertex, which is the peer on the trust path, $p_0, p_1, \ldots p_n \in P$. There is no repeated vertex on one path, all vertices on one path are uniquely existing.

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

Length of the trust path
The number of the steps s is used to measure the length of the trust path. The number of steps between two peers on one trust path is the number of edges between two vertices . For example (see Figure 4.1), the number of steps from peer A to peer D through the path $A - B - C - D$ is $s = 3$.

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 rest peers on the same trust path are indirect friends of the source peer.

Trust decay
As described above, the trust relationship is transitive via the directed trust path. However, as the trust path becomes longer, the trust attitude should decrease gradually between the initial peer and the very end peer. 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 as *trust decay*. The value of trust decay is between 0 and 1. For all of peers in the system, the trust decay has:

$$D : P \times P \rightarrow [0, 1]. \tag{4.6}$$

We define that the trust decay from the source peer to each direct friend is 1. When a third peer is added by the end of the trust path, the trust decay from the source peer to the third one will be $(1 - d)$, where $d \in [0, 1]$ is the decay constant. Accordingly, the trust decay for each further step is decreasing simultaneously as the trust path spreads further. For this reason, 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:

$$D[p_i, p_j] = (1 - d)^{s-1}; \quad p_i, p_j \in P, \tag{4.7}$$

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

The main reason for utilizing the trust concept is to avoid the worst case occurring on the trust path, for which 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 the every end of the target peer will 1 without considering trust decay. In fact, it can not represent the real trust tendency among people in the community. By integrating trust decay into the trust computation, this worst cast will not happen.

Moreover, introducing trust decay is able to let closer friends have more influence on the recommendations, because their suggestions are likely to be more appreciated by the source peer. Meanwhile slightly reduce the order of recommended sights from indirect peers for the psychological trust reason.

Trust network
Trust network graph N contains all directed trust paths belong all peers. Each individual peer has one particular personal trust network graph N_i, which is the sub network of N, and corresponding to the social community of the peer. One personal trust network N_i consists of all directed trust paths from the source peer to every other peers in the peer group. As a result, all paths in one personal trust network always start at the same initial vertex, the source peer. By expanding the trust network, the source peer can connect to the unknown peers, and the peer group gets growing up as well.

Figure 4.2 shows an example of the personal trust network of peer p_a. In this case, the source peer p_a has three direct friends, there are 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 direct trust relationship (they are direct friends), they have already associated with the trust score, and the dash line means two peers are indirectly connected (they are indirect friends), the trust of the source peer to the target peer can be predicted along with the trust path. It also can be observed from the trust network that the source peer can reach the target peer through the different paths. For example, p_a is able to connect to p_g through three different paths. There are $\rho[p_a, p_g]_1 = p_a - p_b - p_g$, $\rho[p_a, p_g]_2 = p_a - p_c - p_b - p_g$ and $\rho[p_a, p_g]_3 = p_a - p_c - p_g$ accordingly. Hence, on the personal trust network, the number of the paths between two peers must have:

$$\forall \rho[p_i, p_j] \geq 1, \tag{4.8}$$

where p_i is the source peer, p_j is one of the members in the p_i's peer group.

The intention of introducing the personal trust network is to find potentially entrusted peers for the source peer, and compute trustworthiness among them.

How to compute trust value and decide the trust path between the source peer and the target peer from a given personal trust network is the main issue of the trust concept. In this thesis, two stages are included.

1. *calculation of Trust on a given path*

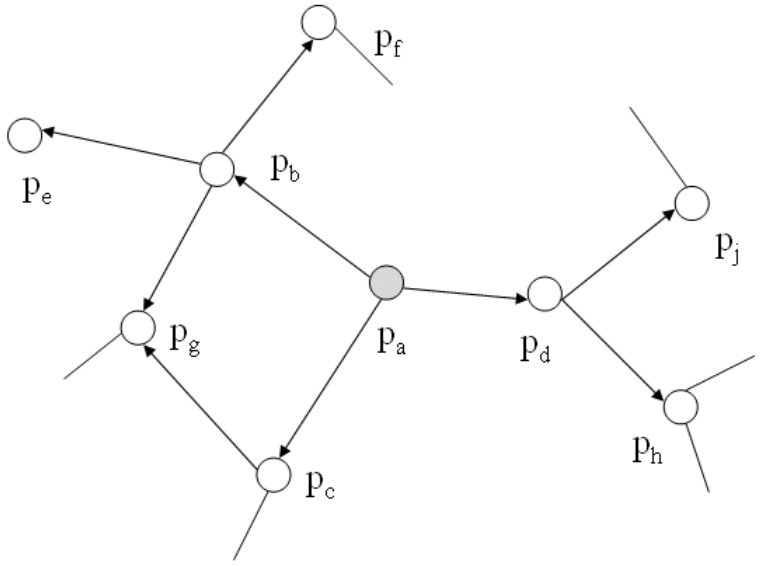

Figure 4.2: p_a's trust network graph

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

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

$$T_p(p_{speer}, p_{tpeer}) = T_{p1} * T_{p2} * \ldots * T_{pn}, \tag{4.9}$$

where n is the number of edges between two peers on a given path, and T_{pi} is the personal trust value associated with the ith edge.

(b) Compute the trust decay from the source peer to the target peer according to the number of the edges between them on the path (see Formula 4.7).

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

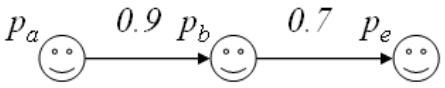

Figure 4.3: Trust transition 1

$$T(p_{speer}, p_{tpeer}) = T_p(p_{speer}, p_{tpeer}) * D[p_{speer}, p_{tpeer}] \quad (4.10)$$

For example, Figure 4.3 shows a directed path from p_a to p_e, two directly connected edges are include in it. Each edge has associated with a personal 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
$D[p_a, p_e] = (1-d)^1$,
if we set the decay constant $d = 0.1$,
$D[p_a, p_e] = (1-0.1)^1 = 0.9$,
the finial computed trust from p_a to p_b is
$T(p_a, p_e) = T(p_a, p_e) * D[p_a, p_e] = 0.567$.

2. *Principle of choosing the trust value*

How to decide a proper trust value of the source peer to the target peer in the situation where several paths exist between them? In this thesis, the over all maximum trust flow is considered as the real trust of the source peer to the target peer. To achieve it, first of all, we need identify all trust paths ($\forall\rho[p_{speer}, p_{tpeer}]$) of the source peer to the target peer from the personal trust network graph, and then compute the trust on each path (Formula 4.10), finally the maximum trust value is chosen as the real trust between them. The resulting formula for choosing the trust from the trust network graph is:

$$T(p_i, p_j) = \underset{\forall\rho[p_i, p_j]}{Max} \{\prod_{k=1}^{w} T_p(p_i, p_j)(1-d)^{s-1}\}, \quad (4.11)$$

where w is the number of the trust paths that connect the source peer to 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, P. Massa and B. Bhattacharjee [16] chose the minimum number of steps needed

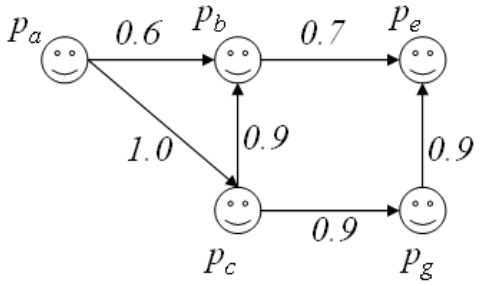

Figure 4.4: Trust transition 2

to reach every other user, they believe the users closest to the current user on the trust network is predicted as more trustable that the users further away. A. Abdul-Rahman and S. Hailes [1] average the trust value over the paths in their trust model.

Figure 4.4 shows an example of the trust network of peer p_a, three paths can be detected 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
$T_{\rho[p_a,p_e]_3} = 1.0 \times 0.9 \times 0.7 \times (1-0.1)^2 = 0.5103$.

It can be seen from the results above, the second trust path $\rho[p_a, p_e]_2 = p_a - p_c - p_g - p_e$ got the highest trust flow, thus the trust of p_a to p_e is 0.6561 in the end.

Confidence

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 preference and judgement. In the real world, people's preferences are different, and this also holds for two most trust friends. For example, a traveler likes go to beach and visit museum (user's preference), he knows that one of his friends is good at surfing and less interested in museum (peer's preference). It is easy to imagine that this friend might give high scores on the surfing places, while assign low scores on museums. If the traveler asks recommendation from this friend on those two subjects, he might have high confidence to the recommendations from this friend on the beach category, and low confidence on the recommendations of the museum

category (user's judgement of the recommended items).

In this thesis, we call the personal confidence on the number of sight groups (the immediate categories of sights) of the friend is the *person-group confidence*, it is displayed as a numeric vector. Each element in the vector is representing a confidence value on a particular sight group, and holds a real value in $[0, 1]$. Thus, the person-group confidence is needed to be customized and subjective, and only can be directly issued and modified by the user to direct acquaintances.

Let $\theta = \{g_1, g_2, \ldots, g_m\}$ be the set containing all sight groups. The corresponding person-group confidences of peers to their direct friends will come out the person-group confidence vectors. They have the form below:

$$C_\theta^p : P \times P \to [0, 1]^m, \tag{4.12}$$

where m is the number of sight groups.

Combining the person-group confidence C_θ^p with the trust T, which is the combination of the personal trust T_p and the trust decay D gained from the trust path, will construct the *confidence vector*. Hence the resulting confidence vector is:

$$C_\theta : P \times P \to [0, 1]^m, \tag{4.13}$$

$$C_\theta = C_\theta^p T = [0, 1]^m \times [0, 1] = [0, 1]^m. \tag{4.14}$$

Table 4.2 is an example showing the way to compute the confidence of p_a to p_d from Figure 4.1. 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, peer p_c has visited all four sight groups. 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 propagated trust decay of 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 confidence vector from p_a to p_c is $\{0.504, 0.4536, 0.1008, 0.252\}$.

Confidence cube

The previous description shows the way to generate a confidence vector for a user to a friend. For all of users in the system, a confidence cube regarding all peers and all sight groups is formulated as:

$$C_{cube} : P \times P \times \theta \to [0, 1]^n \times [0, 1]^m, \tag{4.15}$$

where n is the total number of users in the system, m is the total number of sight groups.

Sight topics	Issued personal 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	$C_{g_1}^{(p_b,p_c)} = 1.0$	$T_p(p_a, p_d) = T_p(p_a, p_b)T_p(p_b, p_c) = 0.9 \times 0.8 = 0.56$	$D(p_a, p_c) = (1 - 0.1)^1 = 0.9$	$C_{g_1}^{(p_a,p_c)} = 0.504$
g_2	$C_{g_2}^{(p_b,p_c)} = 0.9$			$C_{g_2}^{(p_a,p_c)} = 0.4536$
g_3	$C_{g_3}^{(p_b,p_c)} = 0.2$			$C_{g_3}^{(p_a,p_c)} = 0.1008$
g_4	$C_{g_4}^{(p_b,p_c)} = 0.5$			$C_{g_4}^{(p_a,p_c)} = 0.252$

Table 4.1: Example of confidence vector from p_a to p_d

Confidence matrix
Each individual user holds a particular confidence matrix directly from the confidence cube. The confidence matrix of a user can be expressed as:

$$C_{Matrix_{p_i}} : (p_i, P) \times \theta \rightarrow [0, 1] \times [0, 1]^m. \tag{4.16}$$

The confidence matrix is the key parameter for generating trust-based recommendations. By using it, the users can positively influence the recommending procedure on every recommended subject, and eventually have the recommended items under control. However, the user is required to provide more information to the system. A Usability study has shown that users do not mind providing more input to Recommendation system, if they can get better recommendations [13].

Coverage
Coverage is a measure of the number of peers in the user's peer group in the total number of peers in the system. It is shown as:

$$\gamma_{p_i} = \frac{n_{p_i}}{n}, \tag{4.17}$$

where n_{p_i} is the number of peers in the p_i's peer group, and n is the number of peers in the system.

Trust-based recommendation
Trust-based recommendation for a particular user (p_i) R_{p_i} is an ordered list of appreciated nearby sights and a list of trustable peers,

$$R_{p_i} = \{(s_{\lambda_1}, f_1, P_{i_1}), (s_{\lambda_2}, f_2, P_{i_2}), \ldots, (s_{\lambda k}, f_k, P_{i_k})\}, \tag{4.18}$$

where $s_{\lambda_k} \in S_\lambda$ is one of nearby sights, f_k is the computed score given to the sight, $P_{i_k} \subseteq P_i$ is the set of peers who recommended this sight.

Recommendation threshold μ
Recommendation threshold μ is used to display highly recommended sights.

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 user's (p_i) peer group
μ	the threshold of the recommendation
λ	the distance threshold for finding near sights
F	set of all feedbacks of all sights
C	set of all confidence values
T_p	set of all trust values
d	the trust decay of the source peer to the target peer

Table 4.2: Notations used in this thesis

All recommended sights should have the computed scores which are equal or above the threshold μ.

4.3 Trust-based Recommendation generation

In this section, we will represent four approaches of recommending sights to travelers based on the trust concept. In every single description, the enhanced notations with logical operators: ON event IF condition DO action, will be used to state each approach.

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

1. Defining the nearby sight group
   In the real traveling, when a traveler arrives a city or a sight, only surrounding sights are meaningful and useful to the traveler. In addition, in the travel planning [10], a user might similarly requires recommendations close to a given location. In both cases, we have to determine the set of sights near to a given location.

   To find the nearby sights for a given location, we need to get geographic coordinates of the location. GPS device on the mobile is ready to provide precise geographic information. By using it, the neighboring sights can be filtered out from the sight table, the nearby

Method	Description
$nearby(s.location, p.location) \leq \lambda$	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
$AddFriend(p, P_i)$	add peer p_j into the peer group of user p_i
$AddTrust(t, T_{p_i})$	add a trust t into the trust set of user p_i
$ComputeTrust(P_i, p)$	compute the trust of the source peer to the target peer
$update(h, t_p, d)$	update the historic data set by the personal trust and trust decay
$update(h, c, t_p, d)$	update the historic data set by the confidence vector, personal trust and trust decay
$CollectHistoricData(h_p, H_{p_i})$	add historic data of peer p into user's historic data set H_{p_i}
$recommend(H_{p_i}, P_i)$	recommend nearby sights to the user, and provide recommenders to the user
$TCollaborativeFiltering(H_{p_i}, T_{p_i}, P_i)$	generate recommendation using trust-related collaborative filtering algorithm

Table 4.3: Methods used for recommendation generation

sight must have distance which is less than or equal to a predefined distance criterion λ to the current location. The nearby sight set is $S_\lambda = \{s_{\lambda 1}, s_{\lambda 2}, \ldots, s_{\lambda l}\}$, $S_\lambda \subseteq S$.

ON location event $p_i.location$
IF $\exists s \in S : nearby(s.location, p_i.location) \leq \lambda$
DO $\forall s$ (as above): $AddSight(s, S_\lambda)$

2. Describing four various approaches of creating Trust-based recommendation.

   (a) *Generate trust-based and location-aware recommendations from peer group without considering the trust value*
   This approach is involving three steps. The first one is trust propagation in which the user p_i can find trust peer group P_i , the second is to collect historic data of the peer group H_{p_i} regarding nearby sights, and then generate recommendations based on peers' information only.

   The user's peer group P_i contains direct peers only. From their historic travel information, the sights, that the peers have visited

and also in the nearby sight group will be extracted associated with feedback. The extracted data forms a peers' historic data set H_{p_i}.

ON collect historic data from direct peers event $p_i.userid$
IF $\exists p \in P : DirectFriend(p_i, p) \wedge \exists h(p) \in H \wedge \exists h(p).location \in S_\lambda$
DO $\forall p$ (as above), $\forall h(p)$ (as above): $AddFriend(p, P_i)$, $CollectHistoricData(h(p), H_{p_i})$

Before generate recommendation from historic data set H_{p_i}, we need to aggregate data in H_{p_i}. There are two reasons to do it. The first reason is that one peer might have visited one sight more than once, so it is necessary to average the given ratings by the same peer to the same sight. The second is that one sight might be visited by more than one friend, thus the final rating of the sight is the average of feedback issued by all friends on the same sight. Meanwhile store the number of peers who visited the same sight. Finally, the generated recommendation R_{p_i} from H_{p_i} is displayed as an ordered sight list, which contains the nearby sights (by using the recommendation threshold μ, the recommended nearby sights should have the average feedbacks which are equal or greater than μ), the sets of peers who suggested the sights, and the average feedbacks issued to the sights.

ON generate recommendation event H_{p_i}, P_i
IF $H_{p_i} \neq null : P_i \neq null$
DO : $recommend(H_{p_i}, P_i)$

If the user wants to gain recommendations from friends and friends of friends who are indirect friends of the user, the historic data set need to be expanded to include the data from the indirect friends. To achieve it, we need to find out the indirect and entrusted peers to enlarge the peer group of the user.

Following the trust network, the indirect and potentially trustable peers can be found easily, they are the direct friends of the peers who are already in the user's peer group. Always keep the peers unique in the peer group. After including the indirect friends into the peer group, the peer group will get extend from P_i to $P_i^{'}$. It will contain the direct friends and the indirect friends together. Accordingly, the historic data set $H_{p_i^{'}}$ will extend along with $P_i^{'}$, and will involve data both from direct and indirect peers.

Finally, the recommendation $R_{p_i'}$ can be produced from the expanded data set $H_{p_i'}$.

ON collect historic data from more peers event P_i
IF $\exists p \in P : DirectFriend((p_j \in P_i), p) \wedge \exists h(p) \in H \wedge \exists h(p).location \in S_\lambda$
DO $\forall p$ (as above), $\forall h(p)$ (as above): $AddFriend(p, P_i)$, $CollectHistoricData(h(p), H_{p_i})$

Recursively execute the procedure above, the user's peer group will include more peers, similarly the historic data set is growing up as well. As a result, the Trust-based recommendation can be regenerated after analyzing the expanded historic data set.

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

(b) *Generate trust-based and location-aware recommendations along with the trust rating and trust decay*
The first solution attempts to group peers who have similar interests with the user, but it does not consider the trustworthiness between peers. The second approach is trying to integrate trust concept into recommendation generation. This approach involves four steps. First of all is still the trust propagation in order to create a peer group for the user, the second step is to collect historic data set H_{p_i} from peers, the third one is to compute trust value among them and integrate trust information into H_{p_i} to form a data set containing trust information $H^t_{p_i}$, finally, generate recommended sights based on $H^t_{p_i}$.

Same as the first approach, the peer group P_i of the current user contains direct friends only, extract historic traveling data of peers to construct a data set H_{p_i}. And 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. After that, use trust values to update the data set H_{p_i} by multiplying corresponding trust values with the given feedbacks by peers respectively, to result a trust-based historic data set $H^t_{p_i}$. By now, the trust concept has been integrated into peers' data. Consequently, the recommended sights can be generated from $H^t_{p_i}$ (the same solution with the first approach). Thus the resulting recommendation R_{p_i} is

also an ordered list, which contains the recommended sights, the sets of recommenders and the average scores of the sights which are equal or above the threshold μ.

ON collect updated historic data by trust from direct peers event $p_i.userid$
IF $\exists p \in P : DirectFriend(p_i, p) : d = ComputeTrust((p_j \in P_i), p) \rightarrow \exists t_p \in T_p \wedge (t_p.user = p_i \wedge t_p.friend = p) \wedge \exists h(p) \in H \wedge \exists h(p).location \in S_\lambda$
DO $\forall t$ (as above), $\forall h$ (as above): $AddFriend(p, P_i)$, $CollectHistoricData(update(h(p), t_p, d), H_{p_i})$

Following the trust network, the peer group of the user can get expanding from P_i to $P_i^{'}$, to involve direct and indirect peers together. The extracted historic data set $H_{p_i^{'}}$ can also be gained based on $P_i^{'}$. When coming to compute the trust values, the overall maximum trust flows are always chosen as the real trust values of the source peer to the target peers. After updating $H_i^{'}$ by trust values, the recommendation can be generated from the trust-based historic data set $H^t_{p_i^{'}}$. The new regenerated recommendations have integrated the direct and indirect peers' opinions which are represent as the feedbacks of the sights, and the interpersonal trust information which is represent as the trust value together.

ON collect updated historic data by trust from more peers event P_i
IF $\exists p \in P : DirectFriend((p_j \in P_i), p) : d = ComputeTrust((p_j \in P_i), p) \rightarrow \exists t_p \in T_p \wedge (((t_p.user = p_j) \in P_i) \wedge t_p.friend = p) \wedge \exists h(p) \in H \wedge \exists h(p).location \in S_\lambda$
DO $\forall t$ (as above), $\forall h$ (as above): $AddFriend(p, P_i)$, $CollectHistoricData(update(h(p), t_p, d), H_{p_i})$

Recursively execute the procedure above, the user's peer group can include more entrusted peers, and the historic data set can be expanded and updated along with the peer group and the trust. As a result, the Trust-based recommendation will be continuously regenerated from them.

In this approach, we let the user interact with the system by issuing the personal trust to each acquaintance. The personal trust contains the information regarding the recommender whose recommendation the user would like to accept. Consequently the

updated historic data set by the personal trust can be seen as containing the user's individual preference of the recommenders. For the indirect trust peers, the trust decay is used to stand for the distance between the source peer and target peer due to the physiologic trust reason in the real world. Accordingly, the user not only can control the recommending procedure by expanding or narrowing down the size of the historic data set, but also can modify the trust to get the recommendations from the bast trusted peers.

(c) *Generate trust-based and location-aware recommendations combining confidence matrix*
The second solution includes the trustworthiness of peers in the real situation into the recommendation generation process. However the trustworthiness is too coarse to represent the real trust relationship between peers. In fact, one person only trusts some rather than all aspects on the other one else sice each person's preference is unique in the world. To represent this trust circumstance in this thesis, a confidence vector is used to specify this kind of trust. It comes out the third approach.

In the third approach, four steps are needed to accomplish the recommendation process. Firstly, create the peer group P_i of the user; secondly build confidence matrix $C_{Matrix_{p_i}}$; thirdly using confidence matrix to update the extracted historic data set from H_{p_i} to $H^c_{p_i}$; generate recommendation from the confidence-matrix-based historic data set by the end.

The initial peer group P_i of the current user still contains direct friends only. And then each trust of the source peer to the target peer is calculate, find out the confidence vector of the source peer to the target peer, and update confidence vector with the corresponding trust. After that, a confidence matrix $C_{Matrix_{p_i}}$ is constructed for the user by grouping the confidence vectors of all peers together.

Using corresponding confidence value updates the historic data getting from the particular peer on the certain subject, to form a confidence-based historic data set $H^c_{p_i}$. Eventually, the recommendation can be generated from $H^c_{p_i}$.

ON collect updated historic data by confidence matrix from direct peers event $p_i.userid$

IF $\exists p \in P : DirectFriend(p_i, p) : d = ComputeTrust((p_j \in P_i), p) \rightarrow \exists t_p \in T_p \wedge (t_p.user = p_i \wedge t_p.friend = p) \wedge \exists h(p) \in H \wedge \exists h(p).location \in S_\lambda \wedge \forall c \in C \wedge (c.user = p_i \wedge c.friend = p)$
DO $\forall t$ (as above), $\forall h$ (as above), $\forall c$ (as above): $AddFriend(p, P_i)$, $CollectHistoricData(update(h(p), c, t_p, d), H_{p_i})$

If the user expands the peer group to include indirectly entrusted peers, the recommendations will recreate from the expanded confidence-based historic data set $H^c_{p_i}$ below:

ON collect updated historic data by confidence matrix from more peers event P_i
IF $\exists p \in P : DirectFriend((p_j \in P_i), p) : d = ComputeTrust((p_j \in P_i), p) \rightarrow \exists t_p \in T_p \wedge (t_p.user = (p_j \in P_i) \wedge t_p.friend = p) \wedge \exists h(p) \in H \wedge \exists h(p).location \in S_\lambda \wedge \forall c \in C \wedge (c.user = (p_j \in P_i) \wedge c.friend = p)$
DO $\forall t$ (as above), $\forall h$ (as above), $\forall c$ (as above): $AddFriend(p, P_i)$, $CollectHistoricData(update(h(p), c, t_p, d), H_{p_i})$

The third approach not only considers the personal trust, but also the confidence on each different aspect, so the updated historic data set ont only contains the user preferred recommenders, but also contains the user preferred recommended items. According, the generated recommendation from the confidence-based historic data set should be closer to the user's wants, and easier to be accepted than the second approach. However, the computational cost in the recommending process is much higher than the second approach.

(d) *Improve collaborative filtering by trust and location-aware.*
Collaborative filtering tries to find other users who have attitudinal similarities with the current user, and then recommend items that are liked by similar users to the current user. If we do not consider the potential problem about the malicious user, this solution can theoretically offer satisfied recommendations to the user. However, the computational cost is the main weak point of the collaborative algorithm. It will cause huge computation on the system, if the system needs to provide recommendations simultaneously to all n concurrent users in the system , the computational complexity will be N^n.

One way of reducing the unnecessary computation quantity is trying to find a small group of users, which share the attitudinal

similarity with the current user, and then apply collaborative filtering only on this group to work out the recommendations for the user. Thus the main issue of pure collaborative filtering becomes how to efficiently find out a small and similar group of users. This problem has been studied by researchers from different areas. Artificial Intelligent is one of them, such as various clustering algorithms, they can precisely provide similar groups, but those algorithms need to be run off-line because of the low speed of computation, they are not appropriate to be used for providing recommendations on-line.

One possible solution to improve the performance of the collaborative filtering is to introduce the trust concept in it. As we mentioned before, naturally grouped people usually share some similar interests or judgements. We say the peers group is group of people who are similar to the user. It offers possible opportunity to combine trust into collaborative filtering to work out the recommendation on on-line system, which is our forth approach.

The forth approach simply includes four steps to recommend sights. Firstly, find out the trust group from the individual trust network, meanwhile each trust of the source peer to the target peer is calculated. And then collect the historic data of all peers to form a historic data set H_{p_i}. When operating collaborative algorithm on data set H_{p_i}, the correlation coefficient of the source peer to every target peer is computed. Trust will be integrated into correlation coefficient to come out the *trust-related correlation coefficient*. Finally, bring the trust-related correlation coefficient back to collaborative filtering to generate the recommendations for the user.

ON collect updated historic data by trust from direct peers event $p_i.userid$
IF $\exists p \in P : DirectFriend(p_i, p) : d = ComputeTrust((p_j \in P_i), p) \rightarrow \exists t_p \in T_p \wedge (t_p.user = p_i \wedge t_p.friend = p) \wedge \exists h(p) \in H \wedge \exists h(p).location \in S_\lambda$
DO $\forall t$ (as above), $\forall h$ (as above): $AddFriend(p, P_i)$, $AddTrust(t, T_{p_i})$, $CollectHistoricData(h(p), H_{p_i})$

ON generate recommendation event H_{p_i}, T_{p_i}, P_i
IF $H_{p_i} \neq null : T_{p_i} \neq null : P_i \neq null$
DO: $TCollaborativeFiltering(H_{p_i}, T_{p_i}, P_i)$

This approach is able to simply and efficiently find out some rather all of similar peers for the user. Although generated recommendation by this solution might not achieve the same quality with pure collaborative filtering, the cost of looking for similar users and computing the similarity is much lower than pure collaborative filtering.

On the other hand, from the real users' point of view, they know clearly about the information source and the recommenders, they are able to inference the order of recommended items through integrating the personal trust with interpersonal similarity. Hence this solution is more transparent than pure collaborative filtering, and can be regarded as a trade off way to improve the performance of pure collaborative filtering both on the scalability and user's satisfaction.

4.4 Summary

This chapter mentioned three kinds of trusts used for recommendation, and mainly described the personal trust, trust propagation and the trust computation, as well as some terms and definitions related to this topic. Beside, present three Trust-based solutions for generating Trust-based recommendations, and one approach is using trust concept to improve the performance of pure collaborative filtering.

Chapter 5

Implementation

Chapter 6

Evaluation

Chapter 7

Related work

Chapter 8

Conclusion

Bibliography

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

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

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

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

[5] X. Gao. Tip connection to greenstone. In *TIP connection to Greenstone*, 2005.

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

[7] 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.

[8] 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.

[9] 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.

[10] X. Huang. Travel planning. In *Travel Planning*, 2005.

[11] M. S. J. Riegelsberger 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.

[12] S. Junmanee. Advanced recommendation methods in the tip system. In *Advanced Recommendation Methods in the TIP system*, 2005.

[13] 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.

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

[15] 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.

[16] 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.

[17] Y. Michel. Location-aware caching in mobile environments. In *Location-aware Caching in Mobile Environments*, 2005.

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

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

[20] 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.

[21] Y. Wang. Travel concept in tip. In *Travel Concept in TIP*, 2005.

[22] W. Ye. Community-based interaction for tip. In *Community-based interaction for TIP*, 2005.

[23] 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.