Smart Itinerary Recommendation Based on User-Generated GPS Trajectories

Hyoseok Yoon[1], Yu Zheng[2], Xing Xie[2], and Woontack Woo[1]

[1] Gwangju Institute of Science and Technology, Gwangju 500-712, South Korea
{hyoon,wwoo}@gist.ac.kr
[2] Microsoft Research Asia, Beijing 100190, China
{yuzheng,xingx}@microsoft.com

Abstract. Traveling to unfamiliar regions require a significant effort from novice travelers to plan where to go within a limited duration. In this paper, we propose a smart recommendation for highly efficient and balanced itineraries based on multiple user-generated GPS trajectories. Users only need to provide a minimal query composed of a start point, an end point and travel duration to receive an itinerary recommendation. To differentiate good itinerary candidates from less fulfilling ones, we describe how we model and define itinerary in terms of several characteristics mined from user-generated GPS trajectories. Further, we evaluated the efficiency of our method based on 17,745 user-generated GPS trajectories contributed by 125 users in Beijing, China. Also we performed a user study where current residents of Beijing used our system to review and give ratings to itineraries generated by our algorithm and baseline algorithms for comparison.

Keywords: Spatio-temporal data mining, GPS trajectories, Itinerary recommendation.

1 Introduction

Traveling is one of the popular leisure activities people do in their free time. Nevertheless, travelers find it challenging to make the most out of the available yet limited time to have quality travels. Often, travelers do not have the luxury of 'trial-and-error' to find interesting locations and routes which could waste the available time once a wrong visit is made. This calls for the needs of good itinerary for travelers. In a series of attempts to ease the burden, many recommendations techniques are researched, especially for the tourism industry [1]. There are a few available options already. Commercial travel agencies provide a handful itineraries starting and ending in major locations with fixed duration of travel, which forces travelers to adapt to the itineraries, rather than receiving an itinerary based on their needs. Travelers also can ask residents in the region or refer to travel experts through travel web sites for recommendation. By intuition, inexperienced travelers can learn from experienced travel experts and active residents of the region to build a better travel plan.

Z. Yu et al. (Eds.): UIC 2010, LNCS 6406, pp. 19–34, 2010.

This is the exact intuition behind our approach to recommend itineraries, but through the data mining of user-generated GPS trajectories from travel experts and active residents of the region to be explored. The use of user-generated GPS trajectories enables many interesting applications. Figure 1 illustrates an application scenario of our work.

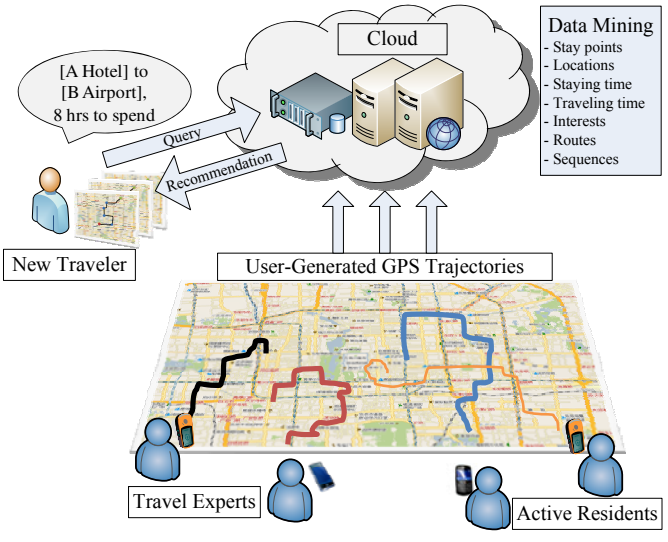

Fig. 1. Application scenario of smart itinerary recommender

As depicted in this application scenario, we recommend new travelers an itinerary that makes the efficient use of the given duration by considering multiple users' accumulated travel routes and experiences. If user-generated GPS trajectories stored on the cloud are accumulated as good examples for data mining, we can extract many features such as where to stay and how long to travel to aid new users in building an efficient travel itinerary.

Our contribution in this paper is as follows.

(1) We propose a *Location-Interest Graph* from multiple user-generated GPS trajectories to model typical user's routes in the area, including which locations are connected and the time relationship between locations.

(2) We model and define what a good itinerary is and how it can be evaluated in order to compare one itinerary to another one.

(3) We present a smart itinerary recommendation framework based on *Location-Interest Graph* generated offline and user query provided online to recommend highly efficient and balanced itinerary that outperforms baseline algorithms.

(4) We evaluate our method using a large GPS dataset collected from 125 users. Then we recommend itineraries from both a large set of simulated user queries and real user inputs to evaluate them according to several characteristics of itinerary and user ratings respectively.

The rest of the paper is organized as follows. Section 2 gives an overview of our system. Section 3 presents detail description of itinerary recommendation processes. In Section 4, we present experiment results and provide discussions. Section 5 reviews related works on itinerary recommendation and GPS data mining, followed by conclusion in Section 6.

2 Overview of Our System

In this section, we first define several terms used throughout the paper. Then we describe the architecture of smart itinerary recommender.

2.1 Preliminaries

Definition 1: Trajectory. A user's trajectory $Traj$ is a sequence of time-stamped points, $Traj = \langle p_1, p_2, ..., p_k \rangle$. Points are represented by $p_i = (lat_i, lng_i, t_i)$, $(i = 1, 2, ..., k)$; t_i is a time stamp, $\forall 1 \leq i < k, t_i < t_{i+1}$ and (lat_i, lng_i) are GPS coordinates of points.

Definition 2: Distance and Interval. $Dist(p_i, p_j)$ denotes the geospatial distance between two points p_i and p_j, and $Int(p_i, p_j) = |p_i.t_i - p_j.t_j|$ is the time interval between two points.

Definition 3: Stay Point. A stay point s is a geographical region where a user stayed over a time threshold T_r within a distance threshold of D_r. In a user's trajectory, s is characterized by a set of consecutive points $P = \langle p_m, p_{m+1}, ..., p_n \rangle$, where $\forall m < i \leq n$, $Dist(p_m, p_i) \leq D_r$, $Dist(p_m, p_{n+1}) > D_r$ and $Int(p_m, p_n) \geq T_r$. Therefore, $s = (lat, lng, t_a, t_l)$, where

$$s.lat = \sum_{i=m}^{n} p_i.lat/|P|, s.lng = \sum_{i=m}^{n} p_i.lng/|P| \tag{1}$$

respectively stands for the average lat and lng coordinates of the collection P; $s.t_a = p_m.t_m$ is the user's arrival time on s and $s.t_l = p_n.t_n$ represents the user's leaving time.

Definition 4: Location History. An individual's location history h is represented as a sequence of stay points they visited with corresponding time of arrival: t_a, time of leave: t_l and time interval from s_i to s_j: $\Delta t_{i,j} = s_j.t_a - s_i.t_l$ where $\forall 1 < i < j \leq n$

$$h = \langle s_1 \overset{\Delta t_{1,2}}{\rightarrow} s_2 \overset{\Delta t_{2,3}}{\rightarrow} s_3, \ \ldots \ , \ s_{n-1} \overset{\Delta t_{n-1,n}}{\longrightarrow} s_n \rangle \tag{2}$$

We put together the stay points detected from all users' trajectories into a dataset $\boldsymbol{S}$, and employ a clustering algorithm to partition this dataset into some clusters. Thus, the similar stay points from various users will be assigned into the same cluster.

Definition 5: Locations. $L = \{l_1, l_2, ..., l_n\}$ is a collection of Locations, where $\forall 1 \leq i \leq n$, $l_i = \{s | s \in \boldsymbol{S}\}$ is a cluster of stay points detected from multiple users' trajectories: $i \neq j$, $l_i \cap l_j = \emptyset$. After the clustering operation, we can substitute a stay point in a user's location history with the cluster ID the stay point pertains to. Supposing $s_1 \in l_i, s_2 \in l_j, s_3 \in l_k, s_{n-1} \in l_l, s_n \in l_m$, Equation (2) can be replaced with

$$h = \langle l_i \overset{\Delta t_{i,j}}{\rightarrow} l_j \overset{\Delta t_{j,k}}{\rightarrow} l_k, \ \ldots \ , \ l_l \overset{\Delta t_{l,m}}{\longrightarrow} l_m \rangle \tag{3}$$

Thus, different users' location histories become comparable and can be integrated to recommend a single location.

Definition 6: Typical Stay Time and Time Interval. For each location $l_i \in L$ with m stay points that pertain to this location, typical stay time t_s of location l_i is defined as median of stay time of stay point $s_k.t_s = s_k.t_l - s_k.t_a$ where $\forall s_k \in l_i$, $\forall 1 \leq k \leq m$.

$$l_i.t_s = Median(s_k.t_s) \tag{4}$$

For n location histories $(h_1, ..., h_n)$ with a sequence $l_i \overset{\Delta t_{i,j}}{\rightarrow} l_j$ where $l_i, l_j \in L$ and $l_i \neq l_j$, typical time interval $\Delta T_{i,j}$ from l_i to l_j is defined as in Equation (5) and all typical time intervals are put into a dataset $\boldsymbol{\Delta T}$ where $\forall 1 \leq k \leq n$.

$$\Delta T_{i,j} = Median(h_k.\Delta t_{i,j}) \tag{5}$$

Definition 7: Location Interest. I_j represents location interest at l_j which has a mutual reinforcement relationship with user travel experience [2]. The mutual relationship of location interest I_j and travel experience e_i are represented as Equation 6 and 7. An item r_{ij} stands for the times that user u_i has stayed in location l_j. Figure 2 depicts this relationship.

$$I_j = \sum\nolimits_{u_i \in U} r_{ji} \times e_i \tag{6}$$

$$e_i = \sum\nolimits_{l_j \in L} r_{ij} \times I_j \tag{7}$$

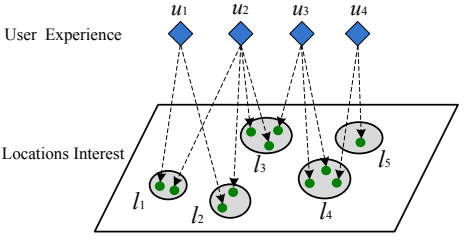

Fig. 2. Location interest and user experience

Definition 8: Trip. A trip $Trip$ is a sequence of locations with corresponding typical time intervals,

$$Trip = \langle l_1 \overset{\Delta T_{1,2}}{\rightarrow} l_2 \overset{\Delta T_{2,3}}{\rightarrow} l_3, \ \ldots \ , \ \overset{\Delta T_{k-1,k}}{\longrightarrow} l_k \rangle \tag{8}$$

where $\forall 1 \leq i < j \leq k$, $\Delta T_{i,j} \in \Delta T$ and $l_i, l_j \in L$ are locations. $Trip$ has four attributes, 1) the total staying time for visiting locations t_{stay}, 2) the total traveling time t_{trav}, 3) the duration of the trip t_{dur} and 4) the interest density of the trip i_{den} defined by the total sum of interest of locations divided by the number of locations.

$$t_{stay} = \sum\nolimits_{i=1}^{k} l_i.t_s \tag{9}$$

$$t_{trav} = \sum\nolimits_{i=1}^{k-1} \Delta T_{i,i+1} \tag{10}$$

$$t_{dur} = t_{stay} + t_{trav} \tag{11}$$

$$i_{den} = (\sum\nolimits_{i=1}^{k} I_i)/k \tag{12}$$

Definition 9: Itinerary. An itinerary It is a recommended trip based on user's start point q_s and destination q_d constrained by trip duration threshold q_t in a query.

$$It = \langle q_s \in l_s \overset{\Delta T_{s,1}}{\rightarrow} l_1 \overset{\Delta T_{1,2}}{\rightarrow} l_2, \ \ldots \ , \ l_{k-1} \overset{\Delta T_{k-1,k}}{\rightarrow} l_k \overset{\Delta T_{k,d}}{\longrightarrow} q_d \in l_d \rangle \tag{13}$$

This means that the user will start a trip from q_s and end in q_d where the duration of trip t_{dur} does not exceed available q_t, $t_{dur} \leq q_t$.

Definition 10: User Query. A user-specified input with three attributes (start point, end point and duration) is defined as a user query, $Q = \{q_s, q_d, q_t\}$.

2.2 Architecture

For the itinerary recommendation, we configure our architecture into offline tasks for processing time-consuming and static information and online tasks for processing variable user queries as depicted in Figure 3. In offline processing, we analyze the user-generated GPS trajectories and build a *Location-Interest Graph* (G_r) with location and interest information, this is quite time consuming process which needs to be done once. Then G_r should be built again only after a significant amount of user-generated GPS trajectories are uploaded. In online processing, we use the G_r built in offline to recommend an itinerary based on a user-specified query.

Our recommendation method is consisted of the following six modular tasks. First two operations, (*Stay Points Extraction and Clustering*, *Location Interest and Sequence Mining*) are carried out in offline and the latter four operations (*Query Verification*, *Trip Candidate Selection*, *Trip Candidate Ranking*, *Re-ranking by Travel Sequence*) are performed online. Details are presented in Section 3.

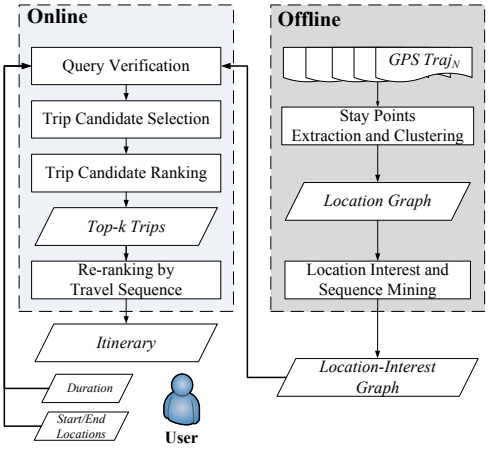

Fig. 3. Architecture of smart itinerary recommender

3 Itinerary Recommendation

In this section, we describe itinerary recommendation processes focusing more on the online processing part. We describe how we model itinerary, how G_r is utilized to generate itineraries and describe the involved selection and ranking algorithms.

3.1 Modeling Itinerary

Since an itinerary is limited by one's available time, known as duration of travel, we use duration as the first constraint in our algorithms. This constraint is very important for two reasons. First reason is that an itinerary with duration that exceeds user's requirement is of no use to users. Second reason is that it simplifies algorithmic complexity by providing a stopping condition. Additionally, we consider the following four factors to determine a good itinerary. We use the following first three characteristics to find trips that surpass some thresholds shown as a cube in Figure 4. The best ideal itinerary would have values equal to 1 in all three dimensions which is depicted as a black dot in Figure 4. The selected trips in the cube are re-ranked according to classical travel sequence to differentiate candidates further.

1) **Elapsed Time Ratio:** An itinerary that uses as much available time as possible is considered to be better, since time is a limited resource for any user, they want to utilize most of their available time.

2) **Stay Time Ratio:** People should spend more time on the locations rather than on the way traveling. An itinerary with less traveling time and more staying time on the site is considered to be a better choice.

3) **Interest Density Ratio:** Visitors to a new region would like to visit as many highly interesting locations as possible, i.e., popular locations and locations with cultural importance.

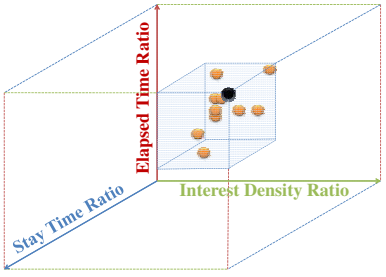

Fig. 4. Trip candidates for a good itinerary

4) **Classical Travel Sequence Ratio:** An itinerary that revisits travel sequences observed from classical travel sequence of previous users is considered to be better, since it reflects realistic routes taken by experts and local people in the region.

3.2 Location-Interest Graph

From multiple users' GPS trajectories, we detect stay points (Definition 3) and cluster them into locations (Definition 5). Further, location interest is calculated (Definition 7) and classical travel sequence is mined by considering hub scores, authority scores and probability of taking this specific sequence (See Section 4.5). Details of mining interesting locations and classical travel sequences are presented in [3]. With this information, we build G_r offline.

Definition 11: Location-Interest Graph (G_r). Formally, a G_r is a graph $G_r = (V, E)$. Vertex set V is Locations (Definition 5) L, $V = L = \{l_1, l_2, ..., l_k\}$. Edge set E is replaced by ΔT where $\Delta T_{i,j}$ stands for a travel sequence from l_i to l_j where $1 \leq i < j \leq k$ with typical time interval as its value. So if there exists an edge between l_i and l_j, then there is a non-zero travel time in corresponding $\Delta T_{i,j}$.

In summary, G_r contains information on 1) Location itself (interest, typical staying time) and 2) relationship between locations (typical traveling time, classical travel sequence).

3.3 Query Verification

In the online process, we first verify user query, $Q = \{q_s, q_d, q_t\}$ by calculating the distance between the start point and end point. There are two approaches we can estimate the distance, $Dist(q_s, q_d)$. First, we can use the harversine formula or the spherical law of cosines with the raw GPS coordinates of start point and end point. Alternatively, we can use Web service such as Bing Map to find traveling distance between two specified locations and traveling time. After confirming the calculated distance with duration in the query, we locate the start point and the end point in G_r by finding the nearest location. Next, the original query $Q = \{q_s, q_d, q_t\}$ is replaced with $Q\prime = \{l_s, l_d, q_t\}$ which is sent to the recommender.

3.4 Trip Candidate Selection

With the verified user query, we select trip candidates from the starting location l_s to the end location l_d. The only restriction we impose in this stage is time constraint so that the candidate trips do not exceed the given duration q_t. We first start from a path which includes the start location l_s as the sole location. Then we check other locations not in this path but are feasible to visit with the remaining duration iteratively. The constraint of duration and visited location information are used as heuristics to select the next location. As we add a new location for the path, we also keep a list of already added locations, so that this location is not checked in the next iteration. For each location added to the path, we subtract the stay time of the location and traveling time to the location to yield a new remaining time. Once the path reaches the end location, we add the generated path as a candidate trip. When all the candidates are added, we return n trip candidates as results.

3.5 Trip Candidate Ranking

Algorithm 1 shows trip candidate ranking algorithm. The algorithm returns an array of $top-k$ trips in decreasing order of the Euclidean distance value.

Algorithm 1. CandidateRanking(G_r,Tr_s,q_t)

Input: A Location-Interest Graph G_r, a set of trips Tr_s, and the duration q_t
Output: A set of top-k trips Tr_r, sorted by Euclidean distance

```
1: for all Trip tr ∈ Tr_s do
2:    for all Location loc ∈ tr do
3:       t_trav ⇐ t_trav + G_r.ΔT_prevLoc,loc
4:       t_stay ⇐ t_stay + loc.t_s
5:       i_den ⇐ i_den + I.loc
6:       prevLoc ⇐ loc
7:    tr.SetTime(t_trav, t_stay)
8:    if tr.i_den > MaxI then
9:       MaxI_d ⇐ tr.i_den
10:   for all Trip tr ∈ Tr_s do
11:      tr.SetEucDist(tr.t_dur/q_t, tr.t_stay/q_t, tr.i_den/MaxI)
12: Tr_r ⇐ SortByEucDist(Tr_s)
13: return Tr_r
```

After selecting n trip candidates from previous step, we rank each trip with factors from Section 3. The factors used to rank each trip $tr_i \in Tr_s$, $1 \leq i \leq k$ are,

1) *Elapsed Time Ratio (ETR)* $= tr_i.t_{dur}/q_t$
2) *Stay Time Ratio (STR)* $= tr_i.t_{stay}/q_t$
3) *Interest Density Ratio (IDR)* $= tr_i.i_{den}/Tr_s.MaxI$

Here, we can use some thresholds value to quickly reject undesirable candidates, i.e., reject candidates with elapsed time ratio less than 0.5. Then we find the Euclidean distance of each trip using these 3 dimensions as in Equation 14. Here

$Tr_s.MaxI$ refers to a maximum interest density value of all candidate trips which we use for normalization. We can assign different weight values for the factors by setting α_1, α_2, and α_3. For our system we treat three factors equally important by setting $\alpha_1 = \alpha_2 = \alpha_3 = 1$.

$$ED = \sqrt{\alpha_1(ETR)^2 + \alpha_2(STR)^2 + \alpha_3(IDR)^2} \tag{14}$$

3.6 Re-ranking by Travel Sequence

We have cut down the number of candidate trips from n to k. These k trips will likely have similar Euclidean distance values. So how can we differentiate between candidates, and recommend one over another? Our solution is to examine each trip's travel sequence and score them for any classical travel sequences.

The classical travel sequence integrates three aspects, the authority score of going in and out and the hub scores, to score travel sequences [3]. Figure 5 demonstrates the calculation of the classical score for a 2-length sequence $l_1 \Rightarrow l_3$. The connected edges represent people's transition sequence and the values on the edges show the times users have taken the sequence. Equation 15 shows the calculation based on the following parts. 1) The authority score of location l_1 (a_{l_1}) weighted by the probability of people moving out from this sequence (Out_{l_1,l_3}). In this demonstration, $Out_{l_1,l_3} = 5/7$. 2) The authority score of location l_3 (a_{l_3}) weighted by the probability of people's moving in by this sequence (In_{l_1,l_3}). 3) The hub scores h_b of the users (U_{l_1,l_3}) who have taken this sequence.

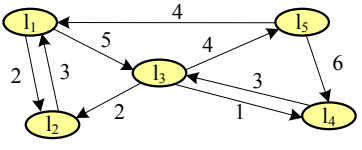

Fig. 5. Demonstration of classical travel sequence

Definition 12: Classical Travel Sequences

$$c_{l_1,l_3} = \sum_{u_k \in U_{l_1,l_2}} (a_{l_1} \times Out_{l_1,l_2} + a_{l_3} \times In_{l_1,l_3} + {h_b}^k) \tag{15}$$

So two trips might have similar value ranges in Euclidean distance after the first ranking, however they will have different classical travel sequence score. We give preference toward trips with higher classical travel sequence score, which means that we recommend trips to revisit previous users' practical travel sequences. Using the classical travel sequence matrix, we can score any travel sequence,

$$c(l_1 \rightarrow l_2 \rightarrow l_3) = c_{1,2} + c_{2,3} \tag{16}$$

Once we have classical travel sequence score of tr_i by calculating $c(tr_i)$, we normalize it by the maximum classical travel sequence score $MaxC$ found of all candidates.

Classical Travel Score Ratio (CTSR) $= c(tr_i)/MaxC$.
Then we once again use the Euclidean distance, this time including classical travel sequence score to re-rank k candidates. We use equal weights for all four factors as shown in Equation 17. The first itinerary with the highest Euclidean distance value is recommended to user, and the user can view alternative itineraries in the order of the Euclidean distance.

$$ED' = \sqrt{\alpha_1(ETR)^2 + \alpha_2(STR)^2 + \alpha_3(IDR)^2 + \alpha_4(CTSR)^2} \tag{17}$$

4 Experiments

In this section, we explain the experiment settings, evaluation approaches, and the experiment results.

4.1 Settings

To collect user-generated GPS trajectories, we have used stand-alone GPS receivers as well as GPS phones. With these devices, 125 users recorded 17,745 GPS trajectories in Beijing from May 2007 to Aug. 2009. In this experiment, time threshold T_r and distance threshold D_r are set to 20 minutes and 200 meters respectively. With these parameters, we detected 35,319 stay points from the dataset and excluded work/home spots. For clustering these stay points into unique locations, we used a density-based clustering algorithm OPTICS (Ordering Points To Identify the Clustering Structure) which resulted in 119 locations. Among these 119 locations, typical traveling time is assigned for the connected locations which serves as an edge set for G_r.

4.2 Evaluation Approaches

In the experiment, we use two evaluation approaches to evaluate our itinerary recommendation methods. First approach is based on a large amount of simulated user queries for the algorithmic level comparison. Using this synthetic data set, we evaluate the quality of the generated itineraries quantitatively compared to other baseline methods. Second approach is based on a user study where the generated itineraries by our method and baselines methods are evaluated by real users. In second approach, we observe how user's perceived quality of itineraries compare by different methods.

Simulation. We used simulation to generate a large quantity of user queries to evaluate the effectiveness of our method. For our simulation to cover most general cases of user input, we used four different levels for duration, 5 hours, 10 hours, 15 hours and 20 hours. Also the duration length seems reasonable for Beijing, China where all the user-generated GPS trajectories are exclusively collected, since it covers an area of about $16,000km^2$. For each duration level, we generated 1,000 queries. Since user query $Q = \{q_s, q_d, q_t\}$ is composed of two points, we generate two sets of GPS coordinates randomly. Here we put some

constraints so that the generated queries follow normal distribution in terms of the distance between the start and end points.

User Study. In user study, we recruited 10 participants who are currently active residents and have lived in Beijing for preferably at least 3 years (average of 3.8 years), since our GPS logs are exclusively collected from the past three years. We asked each participant to use our system to generate itineraries by selecting a start location, an end location and duration of their choice. The recruited participants generated queries in their choice of locations where they were familiar with. Each user submitted 3 queries and gave ratings to 3 itineraries generated by our method and two other baseline methods. They carefully reviewed locations and sequences in the itinerary without knowing about the methods that produced results. Participants took about 30 minutes to completely review 3 sets of 3 itineraries where they were allowed to browse through 3 different itineraries for the query to give relative ratings after comparison. We asked participants following questions to give scores for each generated itinerary in different aspects (score of 1 being the lowest and 5 represents the highest score for better performance) as shown in Table 1.

Table 1. Questions for evaluation

Criteria	Question
Elapsed Time	How efficient is the itinerary in terms of the duration? (1-5)
Stay & Travel Time	How appropriate are staying time and traveling time? (1-5)
Interest	How interesting/representative are the included locations? (1-5)

Baselines. We compared the result of our recommendation with two baseline methods, Ranking-by-Time (RbT) and Ranking-by-Interest (RbI). RbT recommends the itinerary with the highest elapsed time usage. Ideally, it would recommend an itinerary with the elapsed time equal to the duration of the query, if there is such candidate exists. Similarly, RbI ranks the candidates in the order of total interest of locations included in the itinerary. So the candidate with the highest interest density ratio is recommended.

4.3 Results

Simulation. We generated 1,000 queries for each time level (5, 10, 15 and 20) and ran through 3 algorithms. For the duration of 5 hours, only 452 itinerary results were retrieved. For the duration of 10 hours, 15 hours and 20 hours, 935, 961, and 973 itinerary results are acquired respectively. There are three reasons that not all queries returned results. First reason is that simply there was not enough time to go from a start location to an end location. Even though the queries would pass initial query verification, there may be very few or no shorter directions to the end location while consuming the specified duration. Second reason is that there are areas with very few or no locations at all. So when the given time is short and the user starts from one of these sparse areas, the most

of time is used up to go to a nearby location, yielding no results. Third reason is that user starts at a location which has very few outgoing edges, in that case, user might end up in a dead end early even though there are plenty of remaining time. For the recommended itineraries, we looked closely at the average of elapsed time, stay time, interest, classical sequence and Euclidean distance. Figure 6 shows the result for four different time levels. As expected, the baseline algorithm RbT and

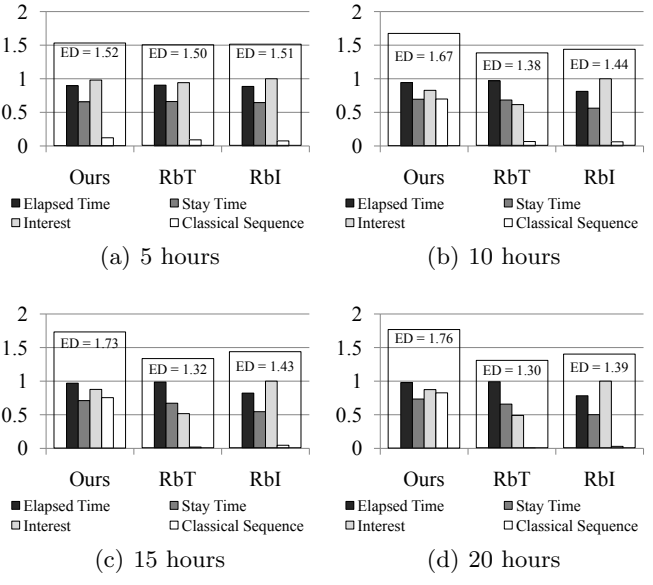

Fig. 6. Simulation results showing the average quality of itinerary generated by different methods

RbI yields best results in the aspect of elapsed time and interest respectively. However, the difference is minimal in the 5 hours level. All three algorithms produced similar quality results. If the duration is very short then there are not many candidates to consider and then many of them would overlap anyway. This explains almost identical graphs in Figure 6(a). The difference gets larger and noticeable as the duration gets longer. Still baseline algorithms successfully recommend itineraries that perform well in only one aspect. RbT has lower average of interest score compared to RbI and our algorithm. Also RbI has lower average of elapsed time compared to RbT and ours. Furthermore, both baseline algorithms produce itineraries that are poor in classical sequence aspect. So we can observe on the average, RbT and RbI will produce a biased or skewed itinerary focusing on only one attribute in a long term. On the other hand, our algorithm produces well-balanced itineraries in all four aspects. The Euclidean distance value gives a good indication that our algorithm produces balanced itinerary overall and even the recommended itineraries are comparable in other

factors that are specialized by baseline algorithms. By looking at the Euclidean distance value, we observe that the performance of our algorithm increases with time whereas two baseline algorithms suffer from performance degradation.

User Study. For 10 participants' 30 queries over Beijing area, we observed the balance of different itinerary attributes in our algorithm compared to the baseline algorithms. As we observed from the simulation, our algorithm produces an itinerary that is well-balanced in the four attributes. So in this user study, we show that our algorithm produces results that are nearly equal to the baselines which specialize in a certain single attribute. For instance, we check how our result compares with RbT produced itinerary in terms of elapsed time. stay time and travel time. Since RbT produces results that maximize the time use, we wanted to check whether the difference user perceives is significant compare to our result which produces well-balanced and nearly close result. Table 2 shows the comparison between our algorithm and RbT in terms of time use. As the T-test reveals that there is no significant advantage in perceived elapsed time, stay time, and travel time from using RbT over ours. Similarly, we compared our result in terms of locations interest included in the itinerary as shown in Table 2. Here again the T-test reveals that there is no significant advantage in perceived interest from using RbI over ours.

Table 2. Comparison of temporal attributes and locations interest

Attribute(s)	**Ours**	**Rank-by-Time**	**T-test**
Elapsed Time	**3.97**	3.67	$p > 0.01$
Stay and Travel Time	**3.60**	3.27	$p > 0.01$
Attribute(s)	**Ours**	**Rank-by-Interest**	**T-test**
Interest	**3.27**	2.92	$p > 0.01$

4.4 Discussions

Temporal aspects. The length of duration is an interesting attribute to look at. Many participants used duration between 6 to 12 hours. It supports our initial assumption that people would not have such a long journey and keep them in a manageable size. For shorter duration, the measured quality of itineraries were less for our algorithm based on Euclidean distance of attributes. Conversely, two baseline algorithms produced the best quality at the shorter duration and recommended less efficient itineraries with longer duration. In extreme cases though, it was possible for baseline algorithms such as RbI to recommend an itinerary that only contains a couple of interesting locations without spending all available time. However, since duration was used as a stopping condition in selecting candidates, most recommended itineraries spend good ratio of available time in simulation and in real user queries alike.

Location interest and classical sequence. Our algorithm produced a balanced itinerary with higher classical sequence scores. In algorithmic level, our algorithm showed a great performance advantage in terms of the four attributes

including classical travel sequence. However, in real queries by users it was difficult to measure location interest and classical travel sequences from the recommended itinerary. Even though an itinerary is composed of many locations and sequences, we only asked the participants to give ratings for the overall location interest and classical sequence. So they gave high score for classical travel sequences they could find, and gave lower score for any abnormal sequences that sometimes balances each other out. So this is different from our simulation where each location interest and classical travel sequences were accumulated to give the overall score. In our current algorithm, we only consider increment of score for location interest and any classical travel sequences found, yet in the real situation, we might need to decrease score or give penalties for totally uninteresting locations and awkward sequences.

5 Related Work

5.1 Itinerary Recommendation

Previously a number of itinerary generation and recommendation systems are introduced. There are interactive systems such as INTRIGUE [4] which provides an interface to browse different categories of location and select an area on a map. In this system, user needs to specify general constraints such as time constraints and user can interactively specify attraction items to be included in the itinerary. Another interactive system is TripTip [5] where a user selects first location to get recommendation on similar types of places using popular tags. Huang and Bian [6] build a travel recommendation system that integrates heterogeneous online travel information based on tourism ontology and recommends tourist attractions using travel preference estimated by the Bayesian network. Kumar et al., [7] present GIS-based Advanced Traveler Information System (ATIS) for Hyderabad City in India which includes a site-tour module based on the shortest distance. Compared to these works, we use simplified query composed of two points and duration where a complete set of itinerary is automatically generated based on real user-generated GPS trajectories. We also present the Euclidean distance based method to compare and measure the quality of itinerary.

5.2 GPS Data Mining Applications

The number of research and applications using GPS data is rapidly increasing. For example many researchers are finding patterns in GPS trajectory [8], convert raw GPS to routable road map [9], use GPS to find locations of interest [10] and combine with multimedia such geo-tagged photo for recommendation [11]. We also previously used GPS data to mine user similarity, interesting locations and travel sequences [2][3], recommend travels [12] and understand user's mobility on transportation mode [13] to understand user and build social networks in GeoLife [14]. Our work in this paper, extends location level recommendation to an itinerary level recommendation and proposes an efficient itinerary recommendation algorithm considering a multiple number of attributes in equally important

weight. We also evaluate our method with a large set of real user-generated GPS trajectories in algorithmic level to the real use cases.

6 Conclusion

In this paper, user-generated GPS trajectories from 125 users were used to build *Location-Interest Graph* which contains useful location-related information (location, interest, stay time, travel time, classical travel sequence). Such information in a region, specifically Beijing, China is used to recommend an itinerary based on user query. An architecture for recommending an itinerary is proposed to handle user-generated GPS trajectories offline and user query processing online. Based on four popular attributes mined from our data set such as elapsed time, staying time, location interest and classical travel sequence, we proposed Euclidean distance based ranking to recommend an itinerary of good quality in all four aspects. We evaluated our method in algorithmic level by using 4,000 simulated user queries in four different time levels to confirm the performance gain in the overall quality over baseline algorithms. We achieved the best performance when it was used for the longer duration. Also active residents used our system to generate real queries and evaluated the resulting itinerary. As a result, we found that our algorithm recommends itineraries that are nearly as good for a single attribute focused baseline algorithms such as Rank-by-Time and Rank-by-Interest and better in overall quality.

Acknowledgments. This research was supported by Microsoft Research Asia and the MKE, Korea, under the ITRC (NIPA-2010-C1090-1011-0008).

References

1. Stabb, S., Werther, H., Ricci, F., Zipf, A., Gretzel, U., Fesenmaier, D., Paris, C., Knoblock, C.: Intelligent systems for tourism. IEEE Intelligent Systems 17(6), 53–66 (2002)
2. Zheng, Y., Zhang, L., Xie, X., Ma, W.Y.: Mining correlation between locations using human location history. In: GIS 2009, pp. 472–475 (2009)
3. Zheng, Y., Zhang, L., Xie, X., Ma, W.Y.: Mining interesting locations and travel sequences from gps trajectories. In: WWW 2009, pp. 791–800 (2009)
4. Ardissono, L., Goy, A., Petrone, G., Segnan, M.: A multi-agent infrastructure for developing personalized web-based systems. ACM Transactions on Internet Technology 5(1), 47–69 (2005)
5. Kim, J., Kim, H., Ryu, J.h.: Triptip: a trip planning service with tag-based recommendation. In: CHI EA 2009, pp. 3467–3472 (2009)
6. Huang, Y., Bian, L.: A Bayesian network and analytic hierarchy process based personalized recommendations for tourist attractions over the Internet. Expert Systems with Applications 36(1), 933–943 (2009)
7. Kumar, P., Singh, V., Reddy, D.: Advanced traveler information system for hyderabad city. IEEE Transactions on Intelligent Transportation Systems 6(1), 26–37 (2005)

8. Monreale, A., Pinelli, F., Trasarti, R., Giannotti, F.: Wherenext: a location predictor on trajectory pattern mining. In: KDD 2009, pp. 637–646 (2009)
9. Cao, L., Krumm, J.: From gps traces to a routable road map. In: GIS 2009, pp. 3–12 (2009)
10. Ashbrook, D., Starner, T.: Using GPS to learn significant locations and predict movement across multiple users. Personal and Ubiquitous Computing 7(5), 275–286 (2003)
11. Tai, C.H., Yang, D.N., Lin, L.T., Chen, M.S.: Recommending personalized scenic itinerary with geo-tagged photos. In: ICME 2008, pp. 1209–1212 (2008)
12. Zheng, Y., Xie, X.: Learning travel recommendation from user-generated gps trajectories. ACM Transaction on Intelligent Systems and Technology 1 (2010) (to be appeared)
13. Zheng, Y., Chen, Y., Li, Q., Xie, X., Ma, W.Y.: Understanding transportation modes based on gps data for web applications. ACM Transactions on the Web 4(1), 1–36 (2010)
14. Zheng, Y., Xie, X., Ma, W.Y.: Geolife: A collaborative social networking service among user, location and trajectory. IEEE Data Eng. Bull. 33(2), 32–39 (2010)