Exploring Pattern-Aware Travel Routes for Trajectory Search

LING-YIN WEI and WEN-CHIH PENG, National Chiao Tung University
WANG-CHIEN LEE, The Pennsylvania State University

With the popularity of positioning devices, Web 2.0 technology, and trip sharing services, many users are willing to log and share their trips on the Web. Thus, trip planning Web sites are able to provide some new services by inferring Regions-Of-Interest (ROIs) and recommending popular travel routes from trip trajectories. We argue that simply providing some travel routes consisting of popular ROIs to users is not sufficient. To tour around a wide geographical area, for example, a city, some users may prefer a trip to visit as many ROIs as possible, while others may like to stop by only a few ROIs for an in-depth visit. We refer to a trip fitting the former user group as an *in-breadth trip* and a trip suitable for the latter user group as an *in-depth trip*. Prior studies on trip planning have focused on mining ROIs and travel routes without considering these different preferences. In this article, given a spatial range and a user preference of depth/breadth specified by a user, we develop a *Pattern-Aware Trajectory Search* (PATS) framework to retrieve the top K trajectories passing through popular ROIs. PATS is novel because the returned travel trajectories, discovered from travel patterns hidden in trip trajectories, may represent the most valuable travel experiences of other travelers fitting the user's trip preference in terms of depth or breadth. The PATS framework comprises two components: *travel behavior exploration* and *trajectory search*. The travel behavior exploration component determines a set of ROIs along with their attractive scores by considering not only the popularity of the ROIs but also the travel sequential relationships among the ROIs. To capture the travel sequential relationships among ROIs and to derive their attractive scores, a user movement graph is constructed. For the trajectory search component of PATS, we formulate two trajectory score functions, the depth-trip score function and the breadth-trip score function, by taking into account the number of ROIs in a trajectory and their attractive scores. Accordingly, we propose an algorithm, namely, *Bounded Trajectory Search* (BTS), to efficiently retrieve the top K trajectories based on the two trajectory scores. The PATS framework is evaluated by experiments and user studies using a real dataset. The experimental results demonstrate the effectiveness and the efficiency of the proposed PATS framework.

Categories and Subject Descriptors: H.2.8 [**Database Management**]: Database Applications—*Data mining, spatial databases*; H.3.3 [**Information Storage and Retrieval**]: Information Search and Retrieval—*Retrieval model*

General Terms: Algorithms, Measurement, Experimentation

Additional Key Words and Phrases: Trajectory pattern mining, trajectory search, route planning, data mining

ACM Reference Format:
Wei, L.-Y., Peng, W.-C., and Lee, W.-C. 2013. Exploring pattern-aware travel routes for trajectory search. ACM Trans. Intell. Syst. Technol. 4, 3, Article 48 (June 2013), 25 pages.
DOI: http://dx.doi.org/10.1145/2483669.2483681

This article is an extended version of Wei et al. [2010], which appeared in *Proceedings of the 1st Workshop on Uncertain Mobile Data Management and Mining*, 372–377.

Authors' addresses: L.-Y. Wei and W.-C. Peng (corresponding author), Department of Computer Science, National Chiao Tung University, No. 1001 University Road, Hsinchu 300, Taiwan; email: wcpeng@gmail.com; W.-C. Lee, Department of Computer Science and Engineering, The Pennsylvania State University, University Park, PA 16802.

1. INTRODUCTION

Due to the wide availability of Global Positioning System (GPS) technology, many portable devices, such as mobile phones, navigation systems, and GPS loggers, are now location aware. Using these GPS-equipped devices, people can log and share their trajectories on the Web [CarWeb 2010; EveryTrail 2009; Bikemap 2010]. Thanks to these technological advances and the availability of trajectory data, research, and applications on trajectory data mining [Jensen et al. 2007; Li et al. 2007; Giannotti et al. 2007; Lee et al. 2007, 2008; Jeung et al. 2008a, 2008b; Zheng et al. 2008], trajectory data management [Wang et al. 2008; Lange et al. 2008; Tian et al. 2009; Cudré-Mauroux et al. 2010], and trajectory search [Chen et al. 2005, 2010; Sherkat and Rafiei 2008] have attracted considerable research efforts in recent years.

Trajectory search is an essential function for many applications and Web sites, for example, EveryTrail [EveryTrail 2009], a trip sharing community. In such Web sites, users are able to share trips and connect with other travelers. By specifying a city name (e.g., Taipei) or a spatial range, a user may retrieve all trajectories that go across the spatial range or the city boundary. Trajectory search also exists in other forms. For example, by specifying a sample trajectory as a query, similar trajectories can be retrieved. Recently, an efficient algorithm for finding K trajectories that best connect a given set of Regions-Of-Interest (ROIs) from a trajectory dataset is proposed in Chen et al. [2010]. Nevertheless, while this pioneering work makes significant advances in trajectory search, it is constrained by the assumption that users already have a priori knowledge regarding the ROIs in the planned trajectory before the query is issued. Without this knowledge, the users may have to try out a number of different ROI sets, which is very time consuming. Without making such assumptions, in this article, we propose a trajectory search framework that comprises a number of innovative trajectory mining, ranking, and searching techniques to support efficient trajectory search.

For trip planning, a variety of trip recommendation services have been developed [Takeuchi and Sugimoto 2007; Li et al. 2008; Zheng et al. 2009b; Zheng and Xie 2011; Wei et al. 2010; Cao et al. 2010; Choudhury et al. 2010; Lu et al. 2010]. A significant amount of research efforts have focused on mining ROIs from users' travel data. In Yoon et al. [2010, 2011], travel routes are derived by specifying the start and end points as well as the duration of the trip. The studies in Choudhury et al. [2010] and Lu et al. [2010] provide another way to discover travel routes from geotagged photos shared by users. However, these travel routes only reveal the visiting sequences of ROIs without indicating the detailed routes. Additionally, users may have different preferences regarding how to tour around a city or a scenic area. To tour around a wide geographical area, for example, a city, some users may prefer a trip to visit as many ROIs as possible, while others may like to stop by only a few ROIs for an in-depth visit. We refer to a trip fitting the former user group as an *in-breadth trip* and a trip suitable for the latter user group as an *in-depth trip*. In this article, we aim to develop a service framework to search for trajectories passing through popular ROIs from archived trip trajectories. The trajectory search query is formulated as follows.

Definition 1.1 (*Top K Trajectory Search*). Given a trajectory dataset, a top K trajectory search query, specified by a spatial range and a user preference of depth/breadth, retrieves the top K trajectories, each of which passes through some popular ROIs within the specified spatial range, from the trajectory dataset.

For instance, a user would like to take a tour in Taipei. He/she could specify a spatial range using an online map service to cover Taipei as well as his/her preference for an in-breath or in-depth trip. In this article, we develop a *Pattern-Aware Trajectory Search*

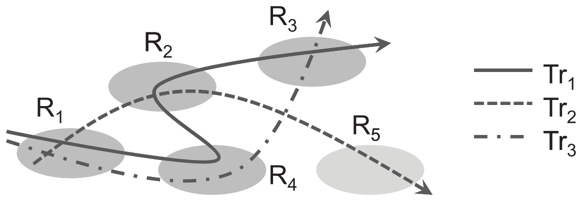


Fig. 1. An example of travel patterns.

(PATS) framework to support the aforementioned trajectory search. As its name indicates, PATS is novel because the returned trajectories, representing the most valuable experiences of other travelers fitting the user's trip preference in terms of breath or depth, are obtained based on discovered travel patterns hidden in trip trajectories. In order to retrieve the top K trajectories that meet in-breadth trips or in-depth trips from a trajectory dataset, some challenging issues (summarized as follows) need to be tackled:

—discovering a set of ROIs and inferring how attractive these ROIs are based on travel patterns hidden in users' trajectories;
—designing trajectory score functions to tailor in-depth or in-breadth trips; and
—developing an efficient trajectory search algorithm to support time-critical online trajectory search.

Our design of the PATS framework comprises two components: (1) travel behavior exploration, and (2) trajectory search. In the travel behavior exploration component, we first extract ROIs, where an ROI is a region if the region has been passed through by a certain number of trajectories. For example, in Figure 1, if we set the threshold of identifying ROIs as 2, we could derive the set of ROIs $\{R_1, R_2, R_3, R_4\}$. As shown in Figure 1, given the set of ROIs, users still have no idea how to plan a trip. For trip planning, a number of ROIs and a travel sequence of ROIs, referred to as a travel route, should be determined. Therefore, the extracted ROIs only reflect the popularity degree of regions, which is not sufficient for trip planning. For example, a travel route, for example, $R_2 \rightarrow R_1 \rightarrow R_4$, may be planned given the set of ROIs. Note that, according to the three trajectories in Figure 1, there is no such travel route which hints that the route may be impracticable. Thus, in this article, we explore *travel patterns* that refer to travel sequential relationships among the ROIs hidden in the trajectory dataset, for trip planning. By aggregating travel sequential relationships among ROIs, travel patterns indicate not only sequential relationships but also transition probabilities among ROIs. By inferring the reachability of an ROI from travel patterns, we could determine the attractive score of the ROI. The reachability of an ROI describes how likely a user would be to pass the ROI when traveling around the region. Consequently, we construct a user movement graph, where each vertex represents an ROI and the edges indicate possible transitions between ROIs. In light of the user movement graph, we propose an Attractive Score (AS) algorithm for ROIs via the Markov process.

In Figure 1, Tr_1 is likely to be an in-breadth trip because it passes through four ROIs. On the other hand, Tr_2 and Tr_3 are likely to be in-depth trips. The two trajectories Tr_2 and Tr_3 have the same number of ROIs. With regard to how to quantify the in-depth degree of trips, we claim that not only the number of ROIs but also the attractive scores of the ROIs should be considered. As a result, for the trajectory search component, we formulate two trajectory score functions, the depth-trip score function (DT) and the breadth-trip score function (BT), by taking into account the number of ROIs in a trajectory and the attractive scores of ROIs. To provide online trajectory search, the response time is critical. Accordingly, we propose a *Bounded Trajectory Search* (BTS) algorithm to efficiently retrieve the top K trajectories based on the two trajectory scores. The main idea of algorithm BTS is to incrementally reduce the searching space to find the top K trajectories without calculating the scores of all the trajectories passing through the user-specified spatial range. To evaluate the proposed PATS framework, we conduct experiments and user studies using a real trajectory dataset. The experimental results show that the PATS framework is able to not only effectively but also efficiently retrieve the top K trajectories consisting of popular ROIs.

The contributions of this article are summarized as follows.

—We propose a new trajectory search framework to support trip planning without requiring prior knowledge of ROIs in the specified spatial range.
—We employ a user movement graph to capture travel patterns hidden in a trajectory dataset and develop an algorithm to determine the attractive scores of the ROIs.
—We design two trajectory score functions DT and BT to tailor in-depth and in-breadth trips, respectively.
—We propose an algorithm BTS for efficiently retrieving the top K trajectories ranked according to DT or BT.
—We perform a comprehensive performance evaluation of PATS by extensive experiments and user studies using a real dataset. The experimental results demonstrate the effectiveness and the efficiency of the proposed PATS.

The rest of the article is organized as follows. In Section 2, related works are presented. In Section 3, the overview of our framework is given. The design of travel behavior exploration is presented in Section 4. In Section 5, the online trajectory search is described. The experimental results are provided in Section 6. Finally, this work is concluded in Section 7.

2. RELATED WORKS

In this section, we first present existing works about trajectory search and then review the research on ROI recommendation. Finally, the existing research works on trip planning are presented.

2.1. Trajectory Search

A considerable amount of research efforts have been put into searching for similar trajectories [Chen et al. 2005; Trajcevski et al. 2007]. An important issue for similar trajectory search is to measure the similarities between the specified trajectory and other trajectories. Thus, most studies focus on the similarity formulation. The authors in Chen et al. [2005] propose an efficient algorithm to discover similar trajectories that have similar geographical movements according to the distance measurement, edit distance on real sequence. Furthermore, by performing translations and rotations on trajectories, the authors in Trajcevski et al. [2007] explore a dynamic similarity measurement in terms of velocity patterns of trajectories and motion transformation. In Chen et al. [2010], the authors formulate a new kind of trajectory search. Given a set of ROIs, the top K trajectories that best connect the given ROIs are derived. The best-connected trajectories are evaluated according to the sum distance between trajectories and a given set of ROIs. The query for the aforementioned trajectory search problem is different from a top K trajectory search query in this article. That is, users do not need to specify a set of ROIs for a top K trajectory query in this article. Our top K trajectory search problem fits some trip planning scenarios in which users may not know the ROIs in advance. Even if users are aware of ROIs, they still require travel routes to indicate visiting sequences among them. However, given a set of ROIs, the number of travel routes that pass these ROIs is huge and it is hard to identify which routes are of interest to the users. In PATS, we not only extract ROIs but also derive travel routes according to a user preference of depth/breadth. These features distinguish our work from others.

2.2. ROI Extraction and Recommendation

Several studies have been developed to extract and recommend ROIs [Simon and Fröhlich 2007; Abowd et al. 1997; Park et al. 2007]. In Simon and Fröhlich [2007], the authors recommend ROIs which are not only close but also visible to users. According to user preferences, ROIs are recommended by a user's historical preferences in Abowd et al. [1997], Park et al. [2007], and Yoon et al. [2009]. Notice that these ROIs are predefined and derived from existing tour guides or travel information. Therefore, some studies further develop approaches to automatically determine ROIs from a set of trajectories. In Kang et al. [2005], Takeuchi and Sugimoto [2007], Giannotti et al. [2007], and Li et al. [2010], the authors develop density-based algorithms to discover ROIs that are frequently visited by users. By observing the stay time of regions, the authors in Hariharan and Toyama [2004] and Li et al. [2008] extract ROIs whose stay time is longer than a predefined time threshold. In addition to the extraction of ROIs, the authors in Zheng et al. [2009b], Zheng and Xie [2011], and Cao et al. [2010] propose an HITS-based algorithm and a PageRank-based algorithm to assign a score to each ROI. The main theme of the preceding research works is just to extract ROIs from a set of trajectories, but they do not discuss how to derive travel routes. Consequently, the problem setting of these works is different from ours.

2.3. Trip Planning

Trip planning problems are studied in Kumar et al. [2005], Gonzalez et al. [2007], Tian et al. [2009], and Yoon et al. [2010, 2011], Choudhury et al. [2010], and Lu et al. [2010]. Given a source and a destination, the studies in Kumar et al. [2005], Gonzalez et al. [2007], and Tian et al. [2009] focus on planning the shortest/fastest trip between a given source and a specified destination. In [Yoon et al. 2010, 2011], the authors target travel route recommendations which should connect the given start and end points within the time duration constraints. Furthermore, in recent years, the authors in Choudhury et al. [2010] and Lu et al. [2010] have aimed at analyzing the geotagged photos from Flickr. They propose other ways to extract ROIs from photo information and to generate travel routes from their analysis of photos. However, these travel routes reveal the visiting sequences of ROIs without indicating detailed routes, and the travel routes are generated without considering a user's preference for breadth/depth. We claim that without a prior knowledge of ROIs, PATS is able to recommend the top K trajectories according to a user's preference.

3. OVERVIEW OF PATS

As presented earlier, in this article, given a spatial range and a user preference of breadth/depth, we aim at deriving the top K trajectories. Note that each trajectory is in fact one of the existing trajectories within or across the specified spatial range, and thus the detailed trip information (i.e., turn-by-turn) for a travel route is available. One challenging task behind PATS is how to derive interest scores of trajectories, and the interest score of a trajectory reflects a user preference of breadth/depth. In PATS, interest scores of each trajectory are determined by the number of ROIs and their corresponding attractiveness. To deal with the aforementioned issue, an offline component, a travel behavior exploration component, is developed. In addition, PATS has a trajectory search component to support online trajectory search. These two components of PATS are detailed as follows.

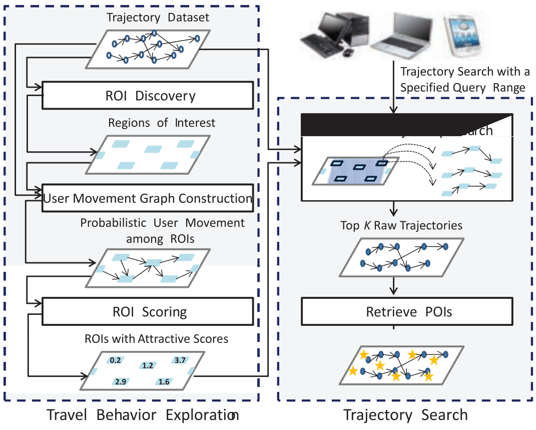
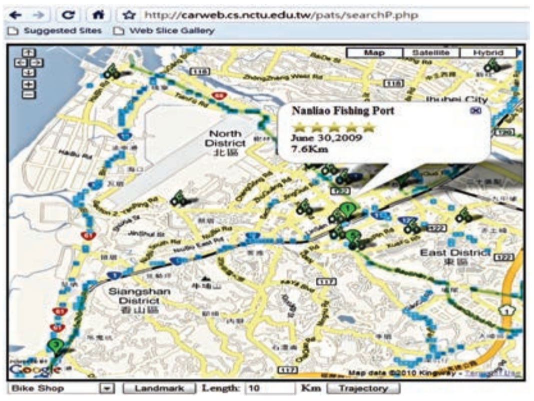
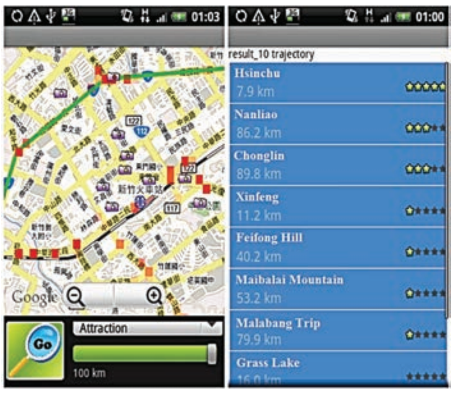

Trajectory Dataset
ROI Discovery
Regions of Interest
User Movement Graph Construction
Probabilistic User Movement among ROIs
ROI Scoring
ROIs with Attractive Scores
0.2
1.2
3.7
2.9
1.6
Travel Behavior Exploration
Trajectory Search with a Specified Query Range
Top K Raw Trajectories
Retrieve POIs
Trajectory Search

Fig. 2. The overview of the proposed framework.





(a) The web interface (b) The Android smart phone interface

Fig. 3. Implementations of our framework.

Travel behavior exploration component. The main task of the travel behavior exploration component is to extract a set of ROIs and to derive their attractive scores. We adopt a density-based approach developed in prior works [Giannotti et al. 2007; Li et al. 2010] to determine ROIs. With the set of ROIs, we consider the travel sequential relationships among the ROIs and propose a user movement graph to capture travel sequential relationships. In the user movement graph, a vertex represents one ROI and edges indicate the transition relationships among ROIs. In light of the user movement graph, we exploit Markov models to assign attractive scores to the ROIs.

Trajectory search component. Through this online component, users issue their queries via our developed Web interface or Android application to retrieve the top K trajectories. Explicitly, Figure 3(a) shows the Web interface, where a user can move/adjust the spatial range on Google map, and the spatial range specified in the Google map is then issued to PATS. In Figure 3(a), the entire spatial range shown in the Web interface, which is marked by the black bounded box, is the spatial range specified. Moreover, the spatial range shown in Figure 3(b) is the input of PATS for our Android application. Once the spatial range and a user preference of depth/breadth are submitted, the top K trajectories and nearby attractions (i.e., points of interest) are displayed on the map.

Clearly, these trajectories are within or across the spatial range issued. For instance, in Figure 3(a), each trajectory is numbered by its rank, and the route of each trajectory is depicted by green lines. For our Android application, the search results of the trajectories are ranked and shown in the right of Figure 3(b). The determination of interest scores of trajectories is discussed later. To efficiently retrieve the top K trajectories, we propose a Bounded Trajectory Search (BTS) algorithm, which is presented in Section 5.

4. DESIGN OF TRAVEL BEHAVIOR EXPLORATION IN PATS

The design of travel behavior exploration in PATS is to determine a set of Regions-Of-Interest (ROIs) with their attractive scores. The travel behavior exploration component is performed in an offline manner. Given a set of trajectories, the travel behavior exploration component first generates ROIs. With the set of ROIs derived, a user movement graph is constructed to capture travel behavior hidden in a trajectory dataset. In light of the user movement graph, we further propose an Attractive Score (AS) algorithm to assign scores to ROIs.

4.1. Determination of Regions-of-Interest

Without loss of generality, a trajectory is expressed by $Tr = <p_1, p_2, \ldots, p_n>$, and each data point at time t_i is denoted as $p_i = (lat_i, lon_i, t_i)$, where the location of data point p_i is represented as a latitude/longitude pair, represented as (lat_i, lon_i). Note that locations of data points derived from GPS have measurement bias or sampling error [Pfoser and Jensen 1999]. To deal with these errors, map-matching methods [Lou et al. 2009; Newson and Krumm 2009] could be applied. In addition, to automatically infer semantics of trajectories, the approaches developed in Yan et al. [2010a, 2011] could be adopted. In this article, each trajectory is shared by users and thus each trajectory reflects a travel route, which indicates a visiting sequence of some landmarks or sightseeing locations. These locations are ROIs and are usually visited by many users. Given a set of trajectories, prior works in Giannotti et al. [2007], Li et al. [2008], Zheng et al. [2009a, 2009b], Kang et al. [2005], and Wei et al. [2010] typically employ a density-based approach to extract ROIs with a high density of passing-by trajectories. Accordingly, we use the density-based approach in Giannotti et al. [2007] to extract ROIs.

The approach in Giannotti et al. [2007] explores grids and uses a density-based concept for the extraction of ROIs. Explicitly, given a set of trajectories and a grid length, the whole geographical space is divided into nonoverlapping grids and the density of each grid is calculated as the number of distinct trajectories passing through it. A trajectory passes through a grid, which means that some data points of the trajectory are located in the grid. If the density of a grid exceeds a given minimum density threshold, the grid is extracted and it is thus identified as one ROI, $\mathfrak{R} = \{R_1, R_2, \ldots, R_m\}$, where each R_i represents one ROI and m is the number of derived ROIs. Once a set of ROIs is derived, a travel route of a trajectory is extracted, where the travel route is represented as a sequence of ROIs. Note that a travel route of a trajectory is regarded as the feature of this trajectory; and for a trajectory Tr, we denote its travel route as $Tr.VS$. To facilitate the presentation of our article, the set of travel routes extracted from the trajectory dataset is called a Travel Route Dataset (TRD).

In reality, the geographical space is divided into a set of functional zones that have different shapes. For example, Figure 4(a) depicts the different functional zones in northern Taiwan, and these functional zones have different shapes. As mentioned before, in our work, the whole space is divided into grids. In Giannotti et al. [2007], nearby grids whose density values are larger than a predefined threshold can be merged into a larger ROI, so it can also be realized in our work. Figure 4(b) shows travel

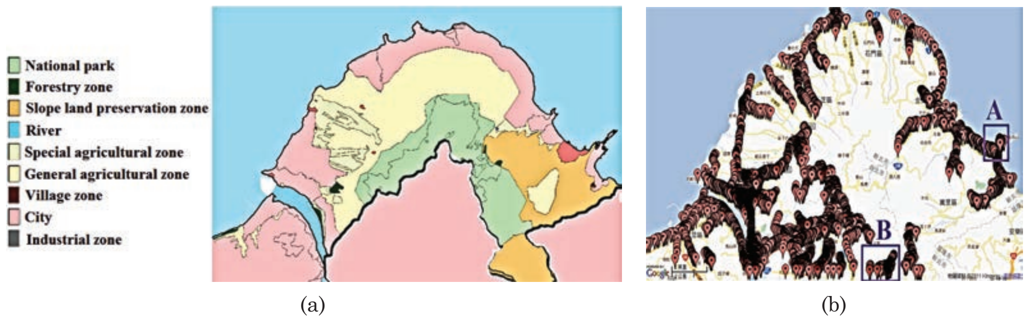

National park
Forestry zone
Slope land preservation zone
River
Special agricultural zone
General agricultural zone
Village zone
City
Industrial zone
A
B
(a)
(b)

Fig. 4. Real trajectory distribution in northern Taiwan and the functional zones in this area [Urban and Rural Development Department, New Taipei City Govement, Taiwan 2010].

trajectories around northern Taiwan. As shown in Figure 4(b), the region A would be merged as an ROI. By referring to Figure 4(a), this ROI belongs to a functional zone (i.e., a national park), which demonstrates that the generation of ROIs could have different shapes. By referring to function zones provided by governments, ROIs that are not related to sightseeing could be filtered out. For example, region B in Figure 4(b) is part of a residential area in a city, which is not a sightseeing region. Consequently, though the space is divided into grids, the determination of ROIs could still discover different shapes via merging grids. Moreover, by referring to function zones in geographical information, ROIs could be refined.

4.2. Construction of User Movement Graph

In our framework, each ROI is associated with an attractive score. If an ROI is likely to be visited by users, it has a higher attractive score. Moreover, not only the popularity of ROIs (that is, the number of user visits) but also the travel behaviors among ROIs should be considered in the formulation of attractive scores. We claim that the attractive scores of ROIs will reflect how likely it is that users will reach the corresponding ROIs from historical travel trajectories shared by users. Intuitively, each trajectory reveals a travel sequential relationship among ROIs. By aggregating travel sequential relationships from a set of trajectories, transition probabilities among ROIs could be derived. Based on the transition probabilities, we could infer the reachability of an ROI. Therefore, a user movement graph is built to capture traversal relationships among ROIs. Different from semantic trajectories [Yan et al. 2010a, 2010b, Yan et al. 2011] developed for high-level representations of trajectories, for example, transportation modes in moves, we focus on exploring travel patterns, that is, travel sequential relationships among the ROIs, hidden in users' travel trajectories in this article.

Given a set of ROIs $\mathfrak{R}$ and a travel route dataset TRD, a user movement graph is a weighted directed graph $UMG = (\mathfrak{R}, E)$, where each vertex represents one ROI and a directed edge $<R_i, R_j>$ exists if there is at least one travel route containing a direct visiting sequence from R_i to R_j. In UMG, each edge is assigned a weight which is the transition probability between two ROIs.

The weight of each edge is derived by aggregating the transition relationships from a set of trajectories. To derive a weight on an edge from R_i to R_j, a possible method is to count the number of pairs $<R_i, R_j>$ in the given travel route dataset and divide it by the total number of $<R_i, R_h>$ for all $R_h \neq R_i$. However, the weight on the edge from R_i to R_j would be mainly influenced by one travel route. For instance, given three travel routes $Tr_1.VS : R_1 \dashrightarrow R_2 \dashrightarrow R_3 \dashrightarrow R_1 \dashrightarrow R_2$, $Tr_2.VS : R_1 \dashrightarrow R_4$, and $Tr_3.VS : R_1 \dashrightarrow R_4$, we can derive the weight on the edge from R_1 to R_2 to be $\frac{1}{2}$ since $<R_1, R_2>$ appears twice in $Tr_1.VS$. Moreover, since $<R_1, R_4>$ also appears in $Tr_2.VS$

Table I. Travel Route Dataset

Tid	Sequence of ROIs (Travel Route)
Tr_1	$R_1 \dashrightarrow R_4 \dashrightarrow R_7$
Tr_2	$R_2 \dashrightarrow R_5 \dashrightarrow R_7 \dashrightarrow R_4 \dashrightarrow R_5 \dashrightarrow R_6$
Tr_3	$R_1 \dashrightarrow R_5 \dashrightarrow R_7$
Tr_4	$R_2 \dashrightarrow R_4 \dashrightarrow R_8$
Tr_5	$R_2 \dashrightarrow R_5 \dashrightarrow R_7$
Tr_6	$R_5 \dashrightarrow R_7 \dashrightarrow R_8 \dashrightarrow R_6$
Tr_7	$R_3 \dashrightarrow R_5 \dashrightarrow R_1$
Tr_8	$R_3 \dashrightarrow R_6 \dashrightarrow R_2$
Tr_9	$R_5 \dashrightarrow R_9 \dashrightarrow R_6$
Tr_{10}	$R_4 \dashrightarrow R_7 \dashrightarrow R_9$

and $Tr_3.VS$, the weight on the edge from R_1 to R_4 is also equal to $\frac{1}{2}$. We argue that these two traversal relationships associated with $<R_1, R_2>$ and $<R_1, R_4>$ should be treated differently because $<R_1, R_4>$ comes from two travel routes. In other words, our framework emphasizes the number of travel routes that have a corresponding traversal relationship. Given a pair of ROIs $<R_i, R_j>$ and a trajectory Tr, if the travel route $Tr.VS$ has a one-step traversal relationship from ROI R_i to ROI R_j, this relationship is contained by Tr, denoted as $<R_i, R_j> \subset Tr$. Since one travel route may have more than one traversal relationship $<R_i, R_j>$, we define an individual out-degree of an ROI from a given trajectory as follows.

Definition 4.1 (*Individual Out-Degree of an ROI with respect to a Given Trajectory*). Given an ROI $R_i \in \mathfrak{R}$ and a trajectory Tr, the individual out-degree of R_i with given Tr is defined as

$$deg^+(R_i|Tr) = |\{R_j|<R_i, R_j> \subset Tr, R_i \neq R_j\}|.$$

For instance, given the travel route dataset in Table I, $deg^+(R_5|Tr_2) = |\{R_j|<R_5, R_j> \subset Tr, R_5 \neq R_j\}| = |\{R_6, R_7\}| = 2$. Accordingly, we define the weights of edges by aggregating the transition relationships from different travel routes in TRD next.

Definition 4.2 (*Weight of a Directed Edge*). Given a travel route dataset TRD and the corresponding user movement graph $UMG = (\mathfrak{R}, E)$, for each directed edge $<R_i, R_j>$ in UMG, the weight of $<R_i, R_j>$ is defined as

$$w_{<R_i,R_j>} = \frac{\sum_{Tr_h \in TRD, <R_i,R_j> \subset Tr_h} \frac{1}{deg^+(R_i|Tr_h)}}{|\bigcup_{R_l \in \mathfrak{R}, R_i \neq R_l} \{Tr_k|Tr_k \in TRD, <R_i, R_l> \subset Tr_k\}|}.$$

Based on Definition 4.2, we judiciously determine weights of edges in a user movement graph. For example, consider the travel route dataset in Table I and a set of ROIs $\mathfrak{R} = \{R_1, R_2, R_3, R_4, R_5, R_6, R_7, R_8, R_9\}$, the directed edge $<R_1, R_4>$ exists since there is at least one trajectory (e.g., Tr_1) whose travel route takes one-step traversing from R_1 to R_4. Since only Tr_1 contains $<R_1, R_4>$, the numerator for the weight of $<R_1, R_4>$ is calculated as

$$\sum_{Tr_h \in TRD, <R_1,R_4> \subset Tr_h} \frac{1}{deg^+(R_1|Tr_h)} = \frac{1}{deg^+(R_1|Tr_1)} = 1.$$

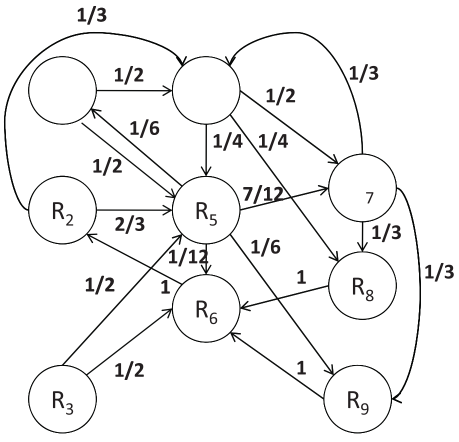


Fig. 5. An example of UMG derived from the travel routes in Table I.

Moreover, since a user at R_1 may possibly move to R_4 and R_5 (based on Tr_1 and Tr_3), the denominator for the weight of $<R_1, R_4>$ is calculated as

$$\left|\bigcup_{R_l \in \Re, R_1 \neq R_l} \{Tr_k | Tr_k \in TRD, <R_1, R_l> \subset Tr_k\}\right| = |\{Tr_1\} \cup \{Tr_3\}| = 2.$$

Consequently, we have $w_{<R_1,R_4>} = \frac{1}{2}$. Given a travel route dataset in Table I, the user movement graph is shown in Figure 5.

4.3. Algorithm AS: Attractive Score for ROIs

In light of the user movement graph, we develop an algorithm to derive the attractive scores of ROIs. According to discovered travel behaviors represented by the user movement graph, we could simulate that how a user traverses among ROIs. For example, given the user movement graph in Figure 5, a user in ROI R_2 is likely to move into ROI R_5 (respectively, R_4) with the probability of $\frac{2}{3}$ (respectively, $\frac{1}{3}$). Since the weights of the edges in the user movement graph indicate transition probabilities among ROIs, we derive the attractive scores of ROIs based on the concept of the Markov process and propose an attractive score algorithm. Similar to the PageRank value for each Web page [Brin and Page 1998], the attractive scores of ROIs are formulated as follows.

Definition 4.3 (*Transition Matrix*). Given a user movement graph $UMG = (\Re, E)$, the transition matrix of UMG is defined as

$$M = \begin{pmatrix} 1-\alpha & \alpha \cdot w_{<R_1,R_1>} & \cdots & \alpha \cdot w_{<R_m,R_1>} \\ 1-\alpha & \alpha \cdot w_{<R_1,R_2>} & \cdots & \alpha \cdot w_{<R_m,R_2>} \\ \vdots & & \ddots & \\ 1-\alpha & \alpha \cdot w_{<R_1,R_m>} & \cdots & \alpha \cdot w_{<R_m,R_m>} \end{pmatrix},$$

where a parameter $\alpha \in [0, 1)$ and $w_{<R_i,R_i>} = 0$ for all $R_i \in \Re$.

Definition 4.4 (*Attractive Score of an ROI*). Given a user movement graph $UMG = (\Re, E)$ and the corresponding transition matrix M, the attractive scores of vertices in

```
ALGORITHM 1: Attractive Score (Algorithm AS)
Input: A travel route dataset, a density threshold $\theta$, a grid length $l$, and a parameter $\alpha$.
Output: Attractive scores of ROIs
/* Preprocessing */
Divide a map into grids with a grid size $l$ by $l$;
Scan trajectories in the raw trajectory dataset to calculate the accumulated density of each grid;
Generate a set of ROIs $\mathfrak{R}$ with density threshold $\theta$;
Merge nearby ROIs whose density is larger than $\theta$;
Generate a travel route dataset $TRD$ by ROIs in $\mathfrak{R}$;
/* User movement graph $UMG(\mathfrak{R}, E)$ generation */
Each element in $\mathfrak{R}$ forms a vertex;
Scan the travel route in $TRD$ to generate edges, weights on edges and the transition matrix $M$;
/* Attractive score */
Let $S^0(R_i) = 1$ for each vertex $R_i \in \mathfrak{R}$;
$j = 0$;
repeat
    $(S^{j+1}(R_1) \cdots S^{j+1}(R_m))^T = M \cdot (1\ \ S^j(R_1)\ \ \cdots S^j(R_m))^T$;
    $j++$;
until $\forall R_i \in \mathfrak{R}$, $S^j(R_i)$ converges;
$\forall R_i \in \mathfrak{R}, S(R_i) = S^j(R_i)$;
return $\{S^j(R_i) | R_i \in \mathfrak{R}\}$;
```

Table II. Attractive Scores of ROIs in Figure 5

	R_1	R_2	R_3	R_4	R_5	R_6	R_7	R_8	R_9
$S(R_i)$	0.36	1.51	0.15	1.11	1.45	1.59	1.34	0.97	0.53

$\mathfrak{R}$ are defined as

$$\begin{pmatrix} S^i(R_1) \\ S^i(R_2) \\ \vdots \\ S^i(R_m) \end{pmatrix} = M \cdot \begin{pmatrix} 1 \\ S^{i-1}(R_1) \\ \vdots \\ S^{i-1}(R_m), \end{pmatrix},$$

where $S^i(R_j)$ is the i-round attractive score of vertex R_j and $S^0(R_j) = 1$ for each $R_j \in \mathfrak{R}$.

Note that the final round attractive score of R_i is represented by $S(R_i)$ for each $R_i \in \mathfrak{R}$.

As shown in Algorithm AS, all ROIs have an initial attractive scores of 1. Then, the attractive scores of the ROIs are calculated iteratively in a round-by-round manner. When the scores are stabilized, which means they are almost the same as the scores derived in the prior round, Algorithm AS stops and the final attractive scores of the ROIs are generated accordingly.

Consider the example in Figure 5. Let $\alpha = 0.85$. In the first round, all the initial attractive scores of the ROIs are set to 1. We iteratively compute the scores of ROIs using Algorithm AS. In this example, the attractive scores of ROIs converge on the twelfth round. The final attractive scores of the ROIs are shown in Table II.

Since algorithm AS is similar to the PageRank algorithm, we can apply the power method [Golub and Loan 1996] to calculate the scores of the ROIs. According to the study Bahmani et al. [1989], the average time complexity to derive the attractive scores of the ROIs that are stabilized is $O(log(n - log\epsilon))$, where n is the number of vertices in the user movement graph $UMG = (\mathfrak{R}, E)$ and $0 < \epsilon < 1$.

5. DESIGN OF TRAJECTORY SEARCH IN PATS

The trajectory search component in PATS provides a search interface for issuing a spatial range K and a user preference depth/breadth. Then, PATS will retrieve the top

K trajectories tailored with a user preference specified. In light of attractive scores of ROIs derived, we formulate two score functions: one for an in-depth trip and the other for an in-breadth trip. Next, we propose a Bounded Trajectory Search algorithm (BTS) to efficiently retrieve the top K trajectories.

5.1. Trajectory Score Functions

In this section, we formulate two score functions to reflect in-depth trips and in-breadth trips. Note that the output of the trajectory search in PATS is the set of top K trajectories. The top K trajectories are those that have higher scores. Clearly, the score of a trajectory will be determined by its corresponding travel route. As mentioned before, a travel route of a trajectory is a visiting sequence of ROIs. Therefore, the score of a trajectory is highly related to not only the number of ROIs but also to the attractive scores of the ROIs.

To measure the scores of a trajectory, two kinds of trip scenarios should be considered: (1) A trip includes a few ROIs with higher attractive scores (i.e., in-depth); (2) a trip includes as many ROIs as possible (i.e., in-breadth). For the in-depth trip, a user may like to go around some ROIs with higher attractive scores even if the number of ROIs is few. The ROIs for an in-depth trip are expected to have a certain level of quality in terms of their average attractive score. Given a spatial range Q, R_Q is the set of ROIs that are within the spatial range Q. Therefore, a trajectory Tr has its Depth-Trip score, denoted as $DT(Tr)$, and $DT(Tr)$ is formulated as

$$DT(Tr) = \frac{1}{|\{R_i|R_i \in Tr.VS \cap R_Q\}|} \sum_{\{R_i|R_i \in Tr.VS \cap R_Q\}} S(R_i).$$

On the other hand, the Breadth-Trip score of a trajectory Tr, denoted as $BT(Tr)$, is defined as

$$BT(Tr) = \sum_{\{R_i|R_i \in Tr.VS \cap R_Q\}} S(R_i).$$

With the preceding formulas, each trajectory is able to have two trajectory scores: depth-trip score and breadth-trip score. If a user would like to have an in-depth trip (respectively, in-breadth trip), our framework will return the top K trajectories according to the depth-trip (respectively, breadth-trip) scores of the trajectories. Note that if a trajectory only has only a few data points within the spatial query range, we only extract these data points of the trajectories (i.e., the subtrajectory) into the search result.

A naive algorithm to perform trajectory search in PATS consists of three steps: (step 1) Given a spatial range Q, the set of ROIs within Q (i.e., R_Q) should be generated first; (step 2) Extract the trajectories whose travel routes contain ROIs in R_Q; and (step 3) Calculate both depth-trip scores and breadth-trip scores for the trajectories derived in step 2. A running example for the naive algorithm is given, where the travel route dataset of trajectories is shown in Table I and the attractive scores of the ROIs is in Table II. Assume that a user issues one spatial range Q and a user preference is set to in-depth trip. As shown in Figure 6, the set of R_Q is $\{R_2\ R_5, R_6, R_7, R_8\}$. Then, in step 2 of the naive algorithm, there are ten candidate trajectories (i.e., $Tr_1, \ldots, Tr_{10}$) selected since the travel routes of these trajectories have ROIs that match some ROIs in R_Q. For each trajectory selected, the depth-trip score is computed. For example, $DT(Tr_3) \approx 1.395$ since ROIs R_5 and R_7 are in $Tr_3.VS \cap R_Q$ (i.e., $DT(Tr_3) = \frac{1}{2}\sum_{R_i \in Tr_3.VS \cap R_Q} S(R_i) = \frac{1}{2}(S(R_5) + S(R_7)) = 1.395$). Similarly, $DT(Tr_2) = \frac{1}{4}\sum_{R_i \in Tr_2.VS \cap R_Q} S(R_i) = \frac{1}{4}(S(R_2) + S(R_5) + S(R_6) + S(R_7)) = 1.48$. The depth-trip scores of the aforesaid 10 trajectories are

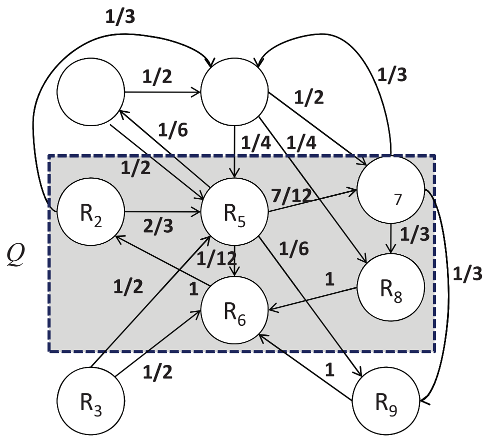


Fig. 6. An illustrative example of a trajectory search.

Table III. Scores of Trajectories by DT in Table I

	Tr_1	Tr_2	Tr_3	Tr_4	Tr_5	Tr_6	Tr_7	Tr_8	Tr_9	Tr_{10}
$DT(Tr_i)$	1.34	$\underline{1.48}$	1.395	1.24	$\underline{1.433}$	1.338	$\underline{1.45}$	$\underline{1.55}$	$\underline{1.52}$	1.34

shown in Table III. If K is set to 5, the top 5 trajectories (i.e., $\{Tr_8, Tr_9, Tr_2, Tr_7, Tr_3\}$) are in the search result.

Since the number of trajectories is huge, the efficiency of the trajectory search is important. Thus, we propose a bounded trajectory search algorithm in PATS.

5.2. Algorithm BTS: Bounded Trajectory Search

The idea of algorithm BTS is to properly maintain score bounds of quantified trajectories and utilize score bounds to eliminate the cost of unnecessary searching for the trajectories that are explicitly not in the top K trajectories.

In PATS, all ROIs are indexed using an R-tree structure. To efficiently retrieve trajectories within ROIs, we implement an inverted trajectory list and for each ROI, and a list of trajectories whose travel routes include that ROI is obtained.

In algorithm BTS, two bounds are maintained: one is the lower bound of the scores (called score bound for simplicity) corresponding to tentatively qualified top K trajectories and the other is the search bound of ROIs to be examined. Given a spatial range Q and a threshold K, we first extract a set of ROIs within Q. Then, these ROIs are sorted by their attractive scores in decreasing order. We denote the sorted list of ROIs as $\mathfrak{R}^s_Q = <R^s_1, \ldots, R^s_h>$, where $S(R^s_i) \geq S(R^s_{i+1})$ for all integer i and $1 \leq i < |R_Q|$. BTS iteratively examines ROIs in the sorted list to retrieve candidates for the top K trajectories and to compute their scores. When an ROI is examined, the corresponding trajectories going through this ROI are retrieved. If the number of the retrieved trajectories is smaller than K, algorithm BTS will select the next ROI from $\mathfrak{R}^s_Q$. On the other hand, if the number of the retrieved trajectories is greater than or equal to K, the score bound is set as the score of the K-th trajectory in the retrieved trajectories. The definition of score bound for the lower bound of the scores corresponding to tentatively qualified trajectories is defined as follows.

Definition 5.1 (*Score Bound for Candidate Trajectories*). Given a set of tentatively qualified candidate trajectories T, the lower bound of the scores (i.e., score bound)

ALGORITHM 2: Bounded Trajectory Search

```
Input: A raw trajectory dataset, a user movement graph $UMG = (\Re, E)$, a spatial range $Q$, and
       a rank-threshold $K$.
Output: Top $K$ trajectories.
for each trajectory $Tr$ in the trajectory dataset do
    A sequence of GPS points of trajectory $Tr$ is transformed into a sequence of ROIs in $\Re$;
    The inverted trajectory lists of corresponding ROIs are generated;
end
$\Re^s_Q \leftarrow$ Sort ROIs in $R_Q$ in decreasing order by attractive scores of ROIs;
Let a set of candidate trajectories $T = \emptyset$;
Let $SBR_T = R^s_h, R^s_i \in \Re^s_Q, i = 1$;
while $R^s_i \leq SBR_T$ do
    $T = T \cup \{Tr | R^s_i \in Tr.AS, \forall Tr \in TTD\}$. //$TTD$: transformed trajectory dataset
    for each trajectory $Tr \in T$ do
        Calculate $DT(Tr)$ (or $BT(Tr)$) with respect to spatial range $Q$;
    end
    Update $LBS_T$, $SBR_T$;
    $i = i + 1$;
end
return Top $K$ trajectories in $T$;
```

of the trajectories in T is defined as $LBS_T = DT(Tr)$ (or $BT(Tr)$), where $DT(Tr)$ (or $BT(Tr)$) is the score at rank K in T.

With the score bound derived LBS_T, algorithm BTS can easily eliminate trajectories whose scores are smaller than LBS_T since these trajectories are not qualified for the top K trajectories. Note that not all ROIs in the sorted list $\Re^s_Q$ should be examined. Thus, the aforementioned search bound for ROIs is used to determine which trajectories should be retrieved as candidates for the top K trajectories. The search bound in ROIs with respect to LBS_T for different trajectory score functions is defined as follows.

Definition 5.2 (*Search Bound for ROIs with respect to* LBS_T). Given a LBS_T and a sequence $\Re^s_Q$, the search bound to be examined for ROIs is defined as

$$SBR_T = R^s_l,$$

where for trajectory score function DT,

$$l = \max\left\{i | S\left(R^s_i\right) \geq LBS_T, \forall i \in \mathbb{N}, 1 \leq i \leq |R_Q|\right\},$$

and for trajectory score function BT,

$$l = \max\left\{i | \sum_{i \leq j \leq |R_Q|} S\left(R^s_j\right) \geq LBS_T, \forall i \in \mathbb{N}, 1 \leq i \leq |R_Q|\right\}.$$

Notice that $\mathbb{N}$ is a set of the natural numbers.

With the aforesaid two bounds, BTS iteratively examines ROIs in $\Re^s_Q$ to retrieve candidates for the top K trajectories until SBR_T is met. For each iteration, LBS_T and SBR_T are updated accordingly. Clearly, the search space for the top K trajectories within the spatial range Q is reduced since not all trajectories passing through Q are retrieved for trajectory score computation, which significantly improves the efficiency of the trajectory search.

Consider the travel route set of trajectories *TRD* in Table I as an example. A spatial range Q (i.e., the rectangle shape in Figure 6) is issued and a threshold K is set

Table IV. Example of Bounded Trajectory Search with Trajectory Score DT

Round	T	LBS_T	SBR_T
Initial	$\emptyset$	∞	R_8
1-round	$\{Tr_8, Tr_9, Tr_2, Tr_6\}$	∞	R_8
2-round	$\{Tr_8, Tr_9, Tr_2, Tr_5, \underline{Tr_6}, Tr_4\}$	1.338	R_7
Final round	$\{Tr_8, Tr_9, Tr_2, Tr_7, \underline{Tr_5}, Tr_3, Tr_6, Tr_4\}$	1.433	R_5

to 5. We can derive $\mathfrak{R}^s_Q = <R_6, R_2, R_5, R_7, R_8>$ by ranking scores in Table II. Note that the attractive scores are derived in the example in Section 4.3. Then, we sequentially scan the ROIs in $\mathfrak{R}^s_Q$ one by one to determine which trajectories are to be selected as candidates for the top 5 trajectories. The detailed rounds of BTS are shown in Table IV. Here, we use DT as our trajectory score function for illustration. In the initial round, the parameters in BTS are set as $T = \emptyset$, $LBS_T = \infty$, and $SBR_T = R_8$. In the first round, algorithm BTS selects R_6 in $\mathfrak{R}^s_Q$ to determine those trajectories that pass through R_6. Via our trajectory invert list, we could have $DT(Tr_2) = 1.48$, $DT(Tr_6) = 1.338$, $DT(Tr_8) = 1.55$, and $DT(Tr_9) = 1.52$. These trajectories are put into T in a decreasing order in terms of their scores (i.e., $T = \{Tr_8, Tr_9, Tr_2, Tr_6\}$). Since the number of trajectories in T is smaller than 5, we need to include more trajectories. Therefore, the next ROI R_2 in $\mathfrak{R}^s_Q$ is selected and the trajectories passing through R_2 are retrieved. Two trajectories (i.e., Tr_4 and Tr_5) pass through R_2 and their scores are computed as $DT(Tr_4) = 1.24$ and $DT(Tr_5) = 1.433$. Hence, we have $T = \{Tr_8, Tr_9, Tr_2, Tr_5, Tr_6, Tr_4\}$. Since $|T| > 5$, the top 5 trajectories in T are determined by their scores. The top 5 trajectories in T are Tr_8, Tr_9, Tr_2, Tr_5, and Tr_6 and the score of the fifth trajectory is $DT(Tr_6) = 1.338$. Accordingly, LBS_T is set to 1.338. Based on the value of LBS_T, SBR_T would be updated. Here, SBR_T is used to shrink the ROI searching space, thereby reducing the search space of candidates for the top 5 trajectories. By Definition 5.2, SBR_T is set to R_7 since the attractive scores of R_6, R_2, R_5, and R_7 are greater than $LBS_T = 1.338$. Note that R_8 is eliminated from the search space since the trajectories passing through the ROIs listed before R_8 in $\mathfrak{R}^s_Q$ should have been examined before R_8 is reached. Yet the remaining trajectories in R_8 do not have DT scores higher than LBS_T and thus are not qualified. Here, we still need to scan the inverted trajectory list of the next ROI R_5 in $\mathfrak{R}^s_Q$. After deriving $DT(Tr_3) = 1.395$ and $DT(Tr_7) = 1.45$, we also update T as $\{Tr_8, Tr_9, Tr_2, Tr_7, Tr_5, Tr_3, Tr_6, Tr_4\}$. Since the fifth trajectory in T is Tr_5, $LBS_T = DT(Tr_5) = 1.433$. Meanwhile, we are able to tighten the search bound (i.e., $SBR_T = R_5$), since the attractive scores of R_6, R_2, and R_5 are greater than $LBS_T = 1.433$. At this point, since SBR_T is reached, the final answer for the top 5 trajectories is derived as $\{Tr_8, Tr_9, Tr_2, Tr_7, Tr_5\}$. The search result is the same as that of the naive algorithm in Section 5.1. In this example, we do not need to retrieve Tr_1 and Tr_{10} for the top K trajectories, reducing the search cost.

Given a user movement graph $UMG = (\mathfrak{R}, E)$ and a trajectory dataset D, the time complexity of the bounded trajectory search algorithm is $O(n(logn + m) + \lambda mlogm)$, where $n = |\mathfrak{R}|$; λ is the number of the scanned ROIs and $\lambda \leq n$; m is the number of candidate trajectories and $m \leq |D|$. In Algorithm BTS, the ROIs which overlap the query range are sorted in decreasing order by their attractive scores and the maximum number of ROIs covered by the query range is n, and thus the time complexity of this step is $O(nlogn)$. Let the number of scanned ROIs be λ and the number of trajectories passing the λ be m. The time complexity for calculating the scores of the m trajectories is $O(mn)$ because each trajectory passes at most n ROIs. To update the score bound for candidate trajectories needs $O(mlogm)$ and it is performed at most λ time, and thus the time complexity is $O(\lambda mlogm)$. Therefore, the overall time complexity is $O(n(logn + m) + \lambda mlogm)$.

5.3. Analysis of Algorithm BTS

In this section, we analyze the correctness of algorithm BTS. The goal of our analysis is to prove that the accuracy of a search result derived by algorithm BTS is good compared to the naive algorithm. In algorithm BTS, by selecting ROIs within a given spatial range, trajectories passing through the selected ROIs are retrieved as candidates for the top K trajectories. The search bound of ROIs to be examined is used as a stop criteria. It is not necessary to retrieve remaining trajectories because trajectories not examined within the search bound are not qualified as the top K trajectories at all. In BTS, we only select trajectories that certainly pass through the ROIs before SBR in $\Re^s_Q$ as candidates for the top K trajectories. If a trajectory only passes through the ROIs after SBR in $\Re^s_Q$ but not any other ROI before SBR in $\Re^s_Q$, this trajectory would not be selected as a candidate for the top K trajectories. In short, all trajectories passing through the spatial range are separated into two sets. The trajectories that only pass through the ROIs after SBR in $\Re^s_Q$, are put in the Noncandidate Set, denoted as NS. The remaining trajectories that certainly go through the ROIs before SBR in $\Re^s_Q$ are put in a Candidate Set, denoted as CS. Hence, in BTS, we only select all trajectories in CS as candidates of top K trajectories. To prove the correctness of BTS, we should prove that there does not exist a trajectory in NS which should be in the top K trajectories. In other words, we prove that the scores of the trajectories in CS are larger than those of the trajectories in NS. The correctness of BTS is proved by Theorem 5.5 and Theorem 5.8 with respect to different trajectory score functions.

In Theorem 5.5, we prove the correctness of BTS with the trajectory score function DT. To prove the correctness of BTS, we prove that the scores of the top K trajectories discovered by BTS are larger than the scores of the trajectories that are not selected as candidates by BTS, that is, the minimum trajectory score of the trajectories in CS must be larger than the maximum trajectory score of the trajectories in NS. Before proving Theorem 5.5, we prove two lemmas used in Theorem 5.5. Lemma 5.3 is used to find out the maximum trajectory score from NS. Lemma 5.4 is used to find out the minimum trajectory score from CS.

In Lemma 5.3, we show how to determine the value of the maximum average of a set of numbers selected from a given set of numbers.

LEMMA 5.3. *Given a set* $\{x_i\}_{i=1}^n$ *where* $x_i > 0$ *for* $1 \le i \le n$ *and* $x_i > x_{i+1}$ *for* $1 \le i < n$, $\max\{\frac{1}{|S'|}\sum_{x\in S'} x | S' \subseteq \{x_i\}_{i=1}^n\} = x_1$.

PROOF. Let $S = \{x_i\}_{i=1}^n$. Suppose all nonempty subsets of S are $S_1^1, S_2^1, \ldots, S_n^1, S_1^2, S_2^2, \ldots, S_{C_2^n}^2, \ldots, S_1^i, \ldots, S_{C_i^n}^i, \ldots, S_1^n$, where S_j^i is a nonempty subset of size i and $1 \le j \le C_i^n$ for $i \in \mathbb{N}$ and $1 \le i \le n$.

Let a nonempty set S_{max}^i be the subset that has the maximum average in S_j^i for $1 \le j \le C_i^n$.

For each $i \in \mathbb{N}$ and $1 \le i \le n$, it is trivial that a nonempty subset S_{max}^i is $\{x_j\}_{j=1}^i$ and the average of the elements in the subset is $\frac{1}{i}\sum_{j=1}^i x_j$.

Then we claim that the average of the elements in S_{max}^i is greater than the average of the elements in S_{max}^{i+1} for $i \in \mathbb{N}$ and $1 \le i < n$. We prove that $\frac{1}{i}\sum_{j=1}^i x_j > \frac{1}{i+1}\sum_{j=1}^{i+1} x_j$ for $i \in \mathbb{N}$ and $1 \le i < n$.

For $i \in \mathbb{N}$ and $1 \le i < n$, since $x_j > x_{i+1}$ for $1 \le j \le i$,

$$\frac{1}{i}\sum_{j=1}^{i} x_j - \frac{1}{i+1}\sum_{j=1}^{i+1} x_j = \frac{1}{i(i+1)}\left((i+1)\sum_{j=1}^{i} x_j - i\sum_{j=1}^{i+1} x_j\right) = \frac{1}{i(i+1)}\left(\sum_{j=1}^{i} x_j - i x_{i+1}\right)$$
$$= \frac{1}{i(i+1)}\sum_{j=1}^{i}(x_j - x_{i+1}) > 0.$$

Due to that $\frac{1}{i}\sum_{j=1}^{i} x_j > \frac{1}{i+1}\sum_{j=1}^{i+1} x_j$ for $i \in \mathbb{N}$, the average of the elements in S^1_{max} is the maximum average and the maximum average is x_1. □

In Lemma 5.4, we show how to determine the value of the minimum average of a set of numbers selected from a given set of numbers.

LEMMA 5.4. *Given a set $\{x_i\}_{i=1}^n$ where $x_i > 0$ for $1 \le i \le n$ and $x_i > x_{i+1}$ for $1 \le i < n$, $\min\{\frac{1}{|S'|}\sum_{x \in S'} x | S' \subseteq \{x_i\}_{i=1}^n\} = x_n$.*

PROOF. The proof is similar to the proof of Lemma 5.3. □

Similarly, Lemma 5.3 and Lemma 5.4 hold even though a part of the distinct elements in set S are equal.

THEOREM 5.5 (CORRECTNESS OF BTS WITH DT SCORE FUNCTION). *Given a sequence of ROIs $\mathfrak{R}^s_Q$, the corresponding set of trajectories T' that pass through at least one ROI in $\mathfrak{R}^s_Q$ and a set of candidate trajectories T with LBS_T and SBR_T, the scores of top K in T are larger than the scores of the trajectories in T'/T.*

PROOF. It is trivial to show that the set of candidate trajectories T is contained in T'. Suppose that there exists a trajectory in T'/T such that the score of the trajectory is greater than the score of one of the top K trajectories in T. Let the size of $\mathfrak{R}^s_Q$ be m and $SBR_T = R^s_h$. If a trajectory in T'/T, the trajectory does not pass through any ROI R^s_i for $1 \le i \le h$, or it will be in T. The reason is that BTS selects trajectories that pass through the ROIs in $\{R^s_1, \ldots, R^s_h\}$. Hence, a trajectory in T'/T only passes through ROIs in $\{R^s_{h+1}, \ldots, R^s_m\}$. We claim that $max\{DT(Tr)|Tr \in T'/T\} = S(R^s_{h+1})$. Since $S(R^s_i) \ge S(R^s_{i+1})$ for each i and $h+1 \le i \le m$, the maximum score of a trajectory that only passes through ROIs in $\{R^s_{h+1}, \ldots, R^s_m\}$ is $S(R^s_{h+1})$ by Lemma 5.3. Similarly, we further claim that the $min\{DT(Tr)|Tr \in T\} = S(R^s_h)$. Since $S(R^s_i) \ge S(R^s_{i+1})$ for each $1 \le i \le h$, the minimum score of a trajectory that only passes through ROIs in $\{R^s_1, \ldots, R^s_h\}$ is $S(R^s_h)$ by Lemma 5.4. Since $S(R^s_h) > S(R^s_{h+1})$, we derive that the maximum trajectory score in T'/T is less than the minimum trajectory score in T. However, the minimum trajectory score is smaller than or equal to the score of the Kth trajectory in T. Then, there does not exist a trajectory in T'/T such that the score of the trajectory is greater than the score of one of the top K trajectories in T. Hence, the scores of the top K trajectories in T are larger than the scores of trajectories in T'/T by contradiction. □

Similarly, we prove the correctness of BTS with the trajectory score function BT in Theorem 5.8. Lemma 5.6 and Lemma 5.7 used in Theorem 5.8 are first proved.

LEMMA 5.6. *Given a set $\{x_i\}_{i=1}^n$ where $x_i > 0$ for $1 \le i \le n$ and $x_i > x_{i+1}$ for $1 \le i < n$, $\max\{\sum_{x \in S'} x | S' \subseteq \{x_i\}_{i=1}^n\} = \sum_{i=1}^n x_i$.*

PROOF. Let $S = \{x_i\}_{i=1}^n$. Suppose all nonempty subsets of S are $S_1^1, S_2^1, \ldots, S_n^1, S_1^2, S_2^2, \ldots, S_{C_2^n}^2, \ldots, S_1^i, \ldots, S_{C_i^n}^i, \ldots, S_1^n$, where S_j^i is a non-empty subset of size i and $1 \le j \le C_i^n$ for $i \in \mathbb{N}$ and $1 \le i \le n$.

Let a nonempty set S_{max}^i be the subset that has the maximum summation in S_j^i for $1 \le j \le C_i^n$.

For each $i \in \mathbb{N}$ and $1 \le i \le n$, it is trivial that a nonempty subset S_{max}^i is $\{x_j\}_{j=1}^i$ and the summation of the elements in the subset is $\sum_{j=1}^i x_j$.

Then, we claim that the summation of the elements in S_{max}^i is smaller than the summation of the elements in S_{max}^{i+1} for $i \in \mathbb{N}$ and $1 \le i < n$. Thus, we prove that $\sum_{j=1}^i x_j < \sum_{j=1}^{i+1} x_j$ for $i \in \mathbb{N}$ and $1 \le i < n$.

For $i \in \mathbb{N}$ and $1 \le i < n$, since $x_{i+1} > 0$ for each i and $1 \le i \le n$,

$$\sum_{j=1}^{i+1} x_j - \sum_{j=1}^{i} x_j = x_{i+1} > 0.$$

$\sum_{j=1}^{i+1} x_j > \sum_{j=1}^i x_j$ for $i \in \mathbb{N}$ holds, showing that the summation of the elements in S_{max}^n is the maximum summation and the maximum summation is $\sum_{i=1}^n x_i$. □

LEMMA 5.7. *Given a set $\{x_i\}_{i=1}^n$ where $x_i > 0$ for $1 \le i \le n$ and $x_i > x_{i+1}$ for $1 \le i < n$, $\min\{\sum_{x \in S'} x | S' \subseteq \{x_i\}_{i=1}^n\} = x_n$.*

PROOF. The proof is similar to the proof of Lemma 5.6. □

THEOREM 5.8 (CORRECTNESS OF BTS WITH *BT* SCORE FUNCTION). *Given a sequence of ROIs $\Re_Q^s$, the corresponding set of trajectories T' that passes through at least one ROI in $\Re_Q^s$ and a set of candidate trajectories T with LBS_T and SBR_T, the scores of top K in T are larger than the scores of the trajectories in T'/T.*

PROOF. The proof for the correctness of BTS with the *BT* score function is similar to the proof of Theorem 5.5 using Lemma 5.6 and Lemma 5.7. □

6. PERFORMANCE EVALUATION

6.1. Dataset and Setting

In this section, we conduct experiments to evaluate the performance of our framework. We crawl travel and biking trajectories around Taiwan from EveryTrail and Bikemap [Bikemap 2010; 2009]. There are 6,548 trajectories and 1,301,192 GPS data points. To extract ROIs, the whole space is divided into grids and the grid length is set to l meters. The impact of grid length will be discussed in our experiments later. A performance study of algorithm BST is investigated. Furthermore, to evaluate the scalability of algorithm BTS, we vary the data size by the number of trajectories in the spatial range (denoted as $|T_Q|$) and the number of ROIs in the spatial range (i.e., $|R_Q|$). The default settings of some important parameters are that the grid length $l = 300$ meters, the rank-threshold $K = 10$, the parameter $\alpha = 0.85$, and the minimum density threshold $\theta = 10$. All experiments are performed on a computer with AMD Phenom II X4 955 processors and 4GB of memory.

6.2. Effectiveness of PATS

In this section, we first conduct experiments to demonstrate trajectories derived under two score functions (i.e., *DT* and *BT*) in PATS. Then, a user study is performed to solicit users' feedback to justify the effectiveness of PATS.

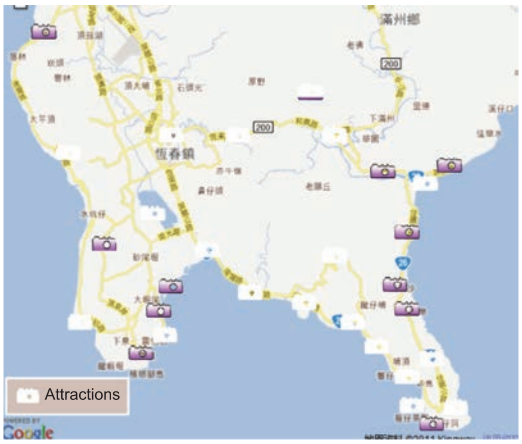


Fig. 7. Attractions in Kenting, Taiwan [Kenting National Park Headquarters, Taiwan 2011].

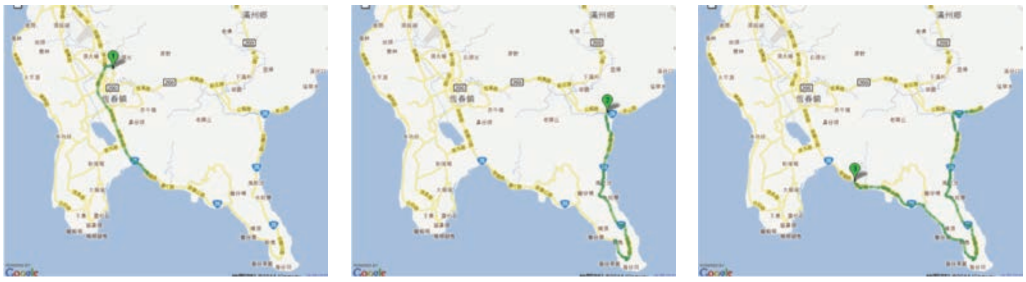
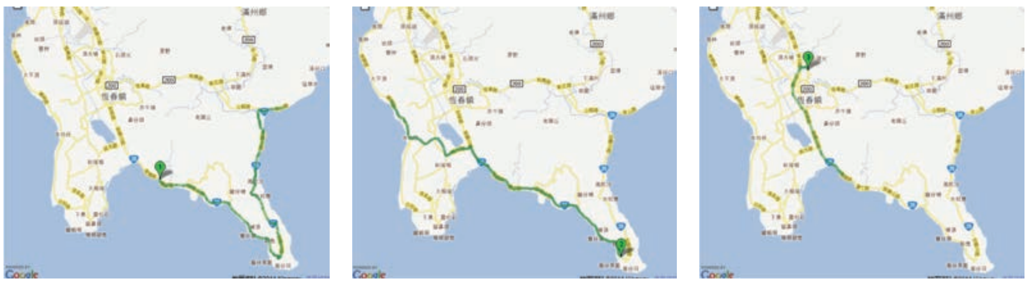


Fig. 8. Top 3 trajectories retrieved under the trajectory score function DT in Kenting.

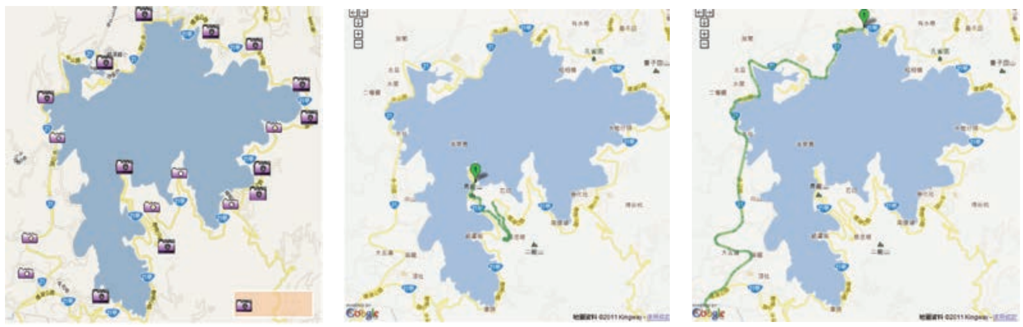
6.2.1. Top K Trajectories under Different Trajectory Score Functions. We first demonstrate the search result of PATS under different trajectory score functions. The spatial range is the range of Kenting, one of the popular travel areas in Taiwan. The attractions in Kenting [Kenting National Park Headquarters, Taiwan 2011] are depicted in Figure 7. Figure 8 (respectively, Figure 9) shows the top 3 trajectories retrieved under the score function DT (respectively, BT) by PATS. As shown in Figure 8(a) and Figure 9(a), the length of the top trajectory ranked by DT is smaller than that of the top trajectory ranked by BT. Clearly, trajectory score function DT includes the ROIs having high attractive scores since the score function DT aims at maximizing the average attractive score of ROIs. With the score function BT, long trajectories, passing through many ROIs, are rank highly in the search results.

6.2.2. Quality Evaluation of Top K Trajectories in PATS. In this section, we solicit users' feedback to evaluate the quality of search results of PATS. In our user study, we randomly invited 30 users from campuses to rate the top K trajectories of PATS. The users consist of 16 females and 14 males, and their ages are between 22 to 35. We provided two search results by issuing spatial ranges: One spatial range covers Kenting and the other is the area of Sun Moon Lake. Due to the fact that we have a sufficient number of trajectories in Kenting and Sun Moon Lake, the search result of these two spatial ranges are sufficient. The spatial ranges for Kenting and Sun Moon Lake are shown in Figure 7 and Figure 10(a), respectively. In our user study, we



(a) rank 1 (b) rank 2 (c) rank 3

Fig. 9. Top 3 trajectories retrieved under trajectory score function BT in Kenting.



(a) attractions in Sun Moon Lake (b) top 1 trajectory under DT (c) top 1 trajectory under BT

Fig. 10. Top-1 trajectory retrieved under different trajectory score functions in Sun Moon Lake.

only set K to be 1. Thus, the top trajectory is retrieved under two score functions. For the spatial range of Kenting scenic area, the top trajectory retrieved for an in-depth trip and the top trajectory retrieved for an in-breadth trip are shown in Figure 8(a) and Figure 9(a), respectively. For the spatial range of Sun Moon Lake scenic area, Figure 10(b) and Figure 10(c) show the top 1 trajectories retrieved for an in-depth trip and an in-breadth trip, respectively. In addition, the attractions in Sun Moon Lake area are illustrated in Figure 10(a).

For each top trajectory under two score functions, we asked the users to rate their interest in the trajectory via four levels: very interested, interested, somewhat interested, or not interested. To provide a quantitative measure, we assigned interest scores to these levels from 3 (very interested) to 0 (not interested). To analyze the effectiveness of the retrieved routes for different travel requirements, the users were divided into two groups: one group of users prefer in-depth trips while the other group of users like in-breadth trips. In the user study, 47% of users preferred in-depth trips (denoted as U_D) and 53% preferred in-breadth trips (denoted as U_B). To evaluate the performance in different travel scenarios, for each retrieved route, we aggregated the rating of each group and averaged the interest scores.

The experimental results of the effectiveness of PATS are presented in Figure 11. As shown in Figure 11(a), given the spatial range in Kenting scenic area, the users in group U_D are indeed more interested in our recommended in-depth trip (i.e., top trajectory derived by the score function DT), because the average interest score of a recommended in-depth trip is higher than the average interest score of a recommended in-breadth trip. In addition, given the query range in Kenting scenic area, the users in group U_B liked our recommended in-breadth trip (i.e., top trajectory retrieved by

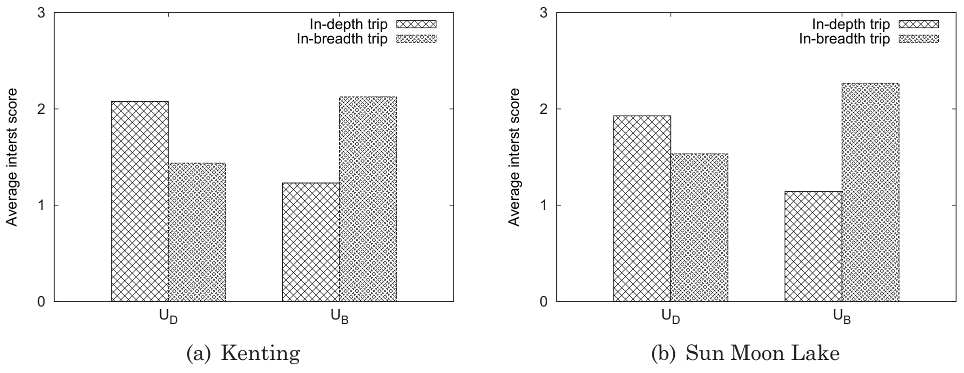


Fig. 11. Quality of trajectories retrieved in PATS for spatial queries in Kenting and Sun Moon Lake.

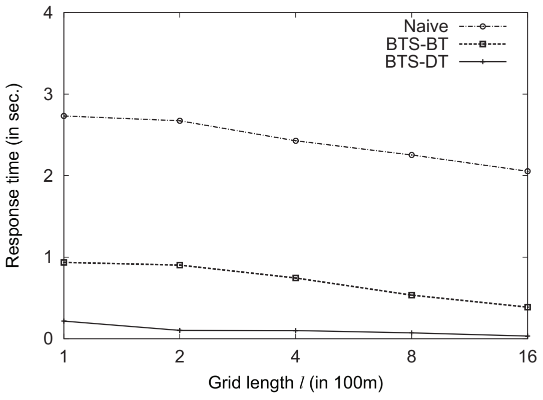


Fig. 12. Effect of different grid lengths.

trajectory score function BT) because the average interest score of a recommended in-breadth trip is higher than the average interest score of a recommended in-depth trip. Similarly, Figure 11(b) shows the result of evaluating the users' feedback for the top trajectory in Sun Moon Lake area. Therefore, the experimental results show that the quality of trajectories derived by PATS is high, and thus PATS could provide different travel routes for two kinds of user preferences.

6.3. Efficiency of Algorithm BTS in PATS

Since algorithm BTS supports online trajectory search in PATS, the efficiency of algorithm BTS is next examined. We compare algorithm BTS with the naive algorithm (algorithm Naive) described in Section 5.1. In the following, the results derived by algorithm BTS with trajectory score functions BT and DT are denoted as BTS-BT and BTS-DT, respectively. We evaluate the efficiency of algorithm BTS by varying grid length l, rank-threshold K, and the trajectory data sizes.

6.3.1. The Impact of Grid Length. In Figure 12, we fix the spatial range such that there is about 1000 trajectories within the spatial range (i.e., $|T_Q| = 1000$) in the experiment. As show in Figure 12, the response time of the three algorithms decreases as the grid length increases. The reason is that the number of ROIs would decrease when the value of grid length l increases. Thus, the computation cost for deriving trajectory scores is decreased due to a smaller number of ROIs. In addition, both BTS-BT and

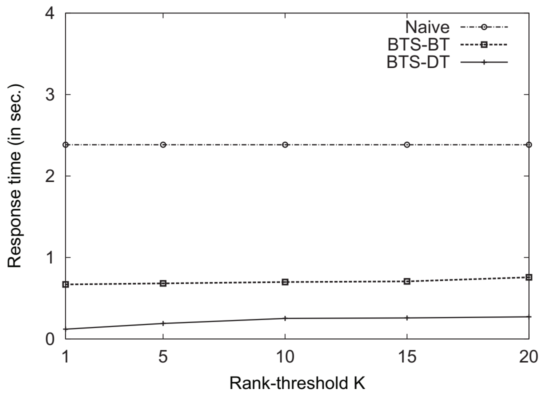
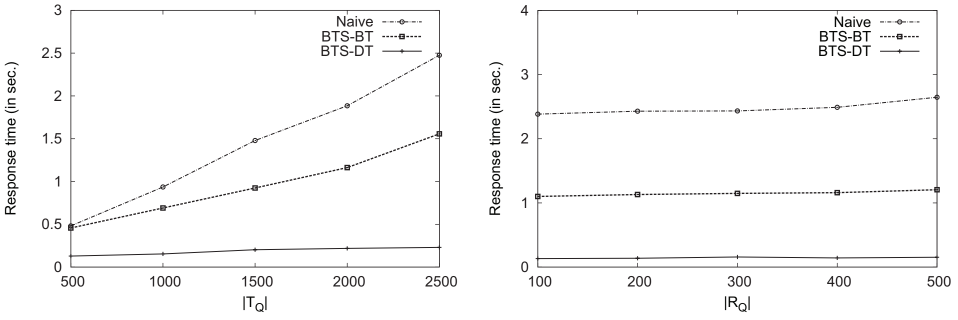


Fig. 13. Effect of different rank-threshold K.



(a) effect of the number of trajectories crossing a spatial range (b) effect of the number of ROIs in a spatial range

Fig. 14. Sensitivity analysis of algorithm Naive and BTS with varied $|T_Q|$ and $|R_Q|$.

BTS-DT outperform algorithm Naive. This shows the advantage of using bounds to reduce search space for the top K trajectories. Although using a larger grid length can improve the efficiency of query time, the discrimination of trajectories is reduced. This is because each trajectory is likely to have the same travel route. Thus, trajectories will have the same trajectory score in terms of depth-trip and breadth-trip scores.

6.3.2. The Impact of Rank-Threshold. Figure 13 shows the effect of K on BTS which discovers the top K trajectories. In the experiment, the spatial range includes 100 ROIs and 1600 trajectories (i.e., $|R_Q| = 100$ and $|T_Q| = 1600$). As can be seen in Figure 13, when the value of K increases, the response time of BTS-BT and that of BTS-DT slightly increases because more candidate trajectories are involved in calculating their trajectory scores. Although the response time of the Naive algorithm is not affected by K, the response time of algorithm Naive is much larger than that of BTS-BT and BTS-DT.

6.3.3. The Impact of Trajectory Data Size. For scalability, two factors affect the performance of online trajectory search. One is the number of ROIs and the other is the number of trajectories. To evaluate the effect of a different number of trajectories (i.e., $|T_Q|$), we randomly choose $|T_Q|$ trajectories that pass through a specified spatial range and vary the number of trajectories (i.e., $|T_Q|$ ranges from 500 to 2500). In addition, we fix the number of ROIs in the spatial range as 500 (i.e., $|R_Q| = 500$) by randomly choosing 500 ROIs within the spatial range. As shown in Figure 14(a), the response time of the three algorithms increases as $|T_Q|$ increases. However, the response times of BTS-DT and BTS-BT are smaller than that of algorithm Naive. Since algorithm BTS is able to reduce the search space for retrieving the top K trajectories, it results in a shorter response time. The performance is prominent when $|T_Q|$ is large.

Similarly, to evaluate the effect of a different number of ROIs, we randomly choose $|R_Q|$ ROIs within the spatial range by varying $|R_Q|$ from 100 to 500. Then, we fix the number of trajectories that pass through the spatial range to be 2000. As shown in Figure 14(b), when $|R_Q|$ increases, the response times of algorithm Naive and BTS-BT (or BTS-DT) slightly increase. This is because that with the increase of $|R_Q|$, the cost of trajectory score calculation would increase. Note that BTS-BT and BTS-DT still have better performance than algorithm Naive. The preceding experimental results show that algorithm BTS in PATS is efficient enough to support online trajectory search.

7. CONCLUSIONS AND FUTURE WORKS

In this article, we have dealt with the problem of searching for pattern-aware travel routes from trajectory datasets. We have developed a framework comprising travel behavior exploration and trajectory search components, to search the top K trajectories with a user preference of depth/breadth. We have built a user movement graph to capture travel behaviors. Then, we have proposed algorithm AS to infer attractive scores of ROIs. In light of attractive scores of ROIs, we have formulated a depth-trip score function and a breadth-trip score function for trajectories. To support time-critical online query, we have proposed algorithm BTS without compromising the accuracy of the search results. We have evaluated our framework using a real dataset and have studied the performance of our framework under different parameter settings. In particular, user studies have been conducted and from these studies, our framework effectively derives interesting travel routes according to user preference of depth/breadth. In addition, the experimental results demonstrate the efficiency of our framework as the query time is less than one second overall. In the future, we will develop more complex travel requirements, such as combining both in-depth and in-breadth patterns and considering the travel time of travel routes. Moreover, for the determination of ROIs, we will explore the attractions from social media using photos and check-in records. With the use of social media, users' travel experiences could be emphasized for travel route planning. In addition, to derive more informative travel routes, we will develop more semantic high-level trajectory search from semantic trajectories.

REFERENCES

ABOWD, G. D., ATKESON, C. G., HONG, J., LONG, S., KOOPER, R., AND PINKERTON, M. 1997. Cyberguide: A mobile context-aware tour guide. *Wirel. Netw. 3,* 5, 421–433.

BAHMANI, B., CHOWDHURY, A., AND GOEL, A. 1989. Statistical complexity of the power method for markov chains. *J. Complex. 5,* 2, 119–143.

BIKEMAP 2010. http://www.bikemap.net.

BRIN, S. AND PAGE, L. 1998. The anatomy of a large-scale hypertextual web search engine. *Comput. Netw. ISDN Syst. 30,* 1–7, 107–117.

CAO, X., CONG, G., AND JENSEN, C. S. 2010. Mining significant semantic locations from gps data. *Proc. VLDB Endow. 3,* 1, 1009–1020.

CARWEB. 2010. http://carweb.cs.nctu.edu.tw.

CHEN, L., OZSU, M. T., AND ORIA, V. 2005. Robust and fast similarity search for moving object trajectories. In *Proceedings of the 25th ACM SIGMOD International Conference on Management of Data (SIGMOD).* 491–502.

CHEN, Z., SHEN, H. T., ZHOU, X., ZHENG, Y., AND XIE, X. 2010. Searching trajectories by locations: An efficiency study. In *Proceedings of the 30th ACM SIGMOD International Conference on Management of Data (SIGMOD).* 255–266.

Choudhury, M. D., Feldman, M., Amer-Yahia, S., Golbandi, N., Lempel, R., and Yu, C. 2010. Automatic construction of travel itineraries using social breadcrumbs. In *Proceedings of the 21st ACM Conference on Hypertext and Hypermedia (HT)*. 35–44.

Cudre-Mauroux, P., Wu, E., and Madden, S. 2010. Trajstore: An adaptive storage system for very large trajectory data sets. In *Proceedings of the 26th International Conference on Data Engineering (ICDE)*. 109–120.

Everytrail 2009. http://www.everytrail.com/.

Giannotti, F., Nanni, M., Pinelli, F., and Pedreschi, D. 2007. Trajectory pattern mining. In *Proceedings of the 13th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (KDD)*. 330–339.

Golub, G. H. and Loan, C. F. V. 1996. *Matrix Computations*. The John Hopkins University Press.

Gonzalez, H., Han, J., Li, X., Myslinska, M., and Sondag, J. P. 2007. Adaptive fastest path computation on a road network: A traffic mining approach. In *Proceedings of the 33rd International Conference on Very Large Data Bases (VLDB)*. 794–805.

Hariharan, R. and Toyama, K. 2004. Project lachesis: Parsing and modeling location histories. In *Proceedings of the 3rd International Conference on Geographic Information Science (GIScience'04)*. 106–124.

Jensen, C. S., Lin, D., and Ooi, B. C. 2007. Continuous clustering of moving objects. *IEEE Trans. Knowl. Data Engin. 19,* 9, 1161–1174.

Jeung, H., Liu, Q., Shen, H. T., and Zhou, X. 2008. A hybrid prediction model for moving objects. In *Proceedings of the 24th International Conference on Data Engineering (ICDE)*. 70–79.

Jeung, H., Yiu, M. L., Zhou, X., Jensen, C. S., and Shen, H. T. 2008. Discovery of convoys in trajectory databases. In *Proceedings of the 34th International Conference on Very Large Databases (VLDB)*. 1068–1080.

Kang, J. H., Welbourne, W., Stewart, B., and Borriello, G. 2005. Extracting places from traces of locations. *ACM SIGMOBILE Mobile Comput. Comm. Rev. 9,* 3, 58–68.

Kenting National Park Headquarters, Taiwan 2011. http://www.ktnp.gov.tw/.

Kumar, P., Singh, V., and Reddy, D. 2005. Advanced traveler information system for hyderabad city. *IEEE Trans. Intell. Transport. Syst. 6,* 1, 26–37.

Lange, R., Durr, F., and Rothermel, K. 2008. Scalable processing of trajectory-based queries in space partitioned moving objects databases. In *Proceedings of the 16th ACM SIGSPATIAL International Conference on Advances in Geographic Information Systems (GIS)*. 1–10.

Lee, J.-G., Han, J., Li, X., and Gonzalez, H. 2008. Traclass: Trajectory classification using hierarchical region based and trajectory based clustering. In *Proceedings of the 34th International Conference on Very Large Databases (VLDB)*. 1081–1094.

Lee, J.-G., Han, J., and Whang, K.-Y. 2007. Trajectory clustering: A partition-and-group framework. In *Proceedings of the 27th ACM SIGMOD International Conference on Management of Data (SIGMOD)*. 593–604.

Li, Q., Zheng, Y., Xie, X., Chen, Y., Liu, W., and Ma, W.-Y. 2008. Mining user similarity based on location history. In *Proceedings of the 16th ACM SIGSPATIAL International Conference on Advances in Geographic Information Systems (GIS)*. 298–307.

Li, X., Han, J., Kim, S., and Gonzalez, H. 2007. Roam: Rule- and motif-based anomaly detection in massive moving object data sets. In *Proceedings of the 7th SIAM International Conference on Data Mining (SDM)*. 273–284.

Li, Z., Ding, B., Han, J., Kays, R., and Nye, P. 2010. Mining periodic behaviors for moving objects. In *Proceedings of the 16th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (KDD)*. 1099–1108.

Lou, Y., Zhang, C., Zheng, Y., Xie, X., Wang, W., and Huang, Y. 2009. Map-matching for low-sampling-rate gps trajectories. In *Proceedings of the 17th ACM SIGSPATIAL International Conference on Advances in Geographic Information Systems (GIS)*. 352–361.

Lu, X., Wang, C., Yang, J.-M., Pang, Y., and Zhang, L. 2010. Photo2trip: Generating travel routes from geotagged photos for trip planning. In *Proceedings of the 18th International Conference on Multimedea (MM)*. 143–152.

Newson, P. and Krumm, J. 2009. Hidden markov map matching through noise and sparseness. In *Proceedings of the 17th ACM SIGSPATIAL International Conference on Advances in Geographic Information Systems (GIS)*. 336–343.

Park, M.-H., Hong, J.-H., and Cho, S.-B. 2007. Location-based recommendation system using bayesian user's preference model in mobile devices. In *Proceedings of the 4th International Conference on Ubiquitous Intelligence and Computing (UIC)*. 1130–1139.

Pfoser, D. and Jensen, C. S. 1999. Capturing the uncertainty of moving-object representations. In *Proceedings of the 6th International Symposium on Advances in Spatial Databases (SSD)*. 111–132.

SHERKAT, R. AND RAFIEI, D. 2008. On efficiently searching trajectories and archival data for historical similarities. In *Proceedings of the 34th International Conference on Very Large Databases (VLDB)*. 896–908.

SIMON, R. AND FROHLICH, P. 2007. A mobile application framework for the geospatial web. In *Proceedings of the 16th International Conference on World Wide Web (WWW)*. 381–390.

TAKEUCHI, Y. AND SUGIMOTO, M. 2007. Cityvoyager: An outdoor recommendation system based on user location history. In *Proceedings of the 3rd International Conference on Ubiquitous Intelligence and Computing (UIC)*. 625–636.

TIAN, Y., LEE, K. C. K., AND LEE, W.-C. 2009. Monitoring minimum cost paths on road networks. In *Proceedings of the 17th ACM SIGSPATIAL International Conference on Advances in Geographic Information Systems (GIS)*. 217–226.

TRAJCEVSKI, G., DING, H., SCHEUERMANN, P., TAMASSIA, R., AND VACCARO, D. 2007. Dynamics-aware similarity of moving objects trajectories. In *Proceedings of the 15th ACM SIGSPATIAL International Conference on Advances in Geographic Information Systems (GIS)*. 1–8.

URBAN AND RURAL DEVELOPMENT DEPARTMENT, NEW TAIPEI CITY GOVEMENT, TAIWAN. 2010. http://www.planning.ntpc.gov.tw/.

WANG, L., ZHENG, Y., XIE, X., AND MA, W.-Y. 2008. A flexible spatio-temporal indexing scheme for large-scale gps track retrieval. In *Proceedings of the 9th International Conference on Mobile Data Management (MDM)*. 1–8.

WEI, L.-Y., PENG, W.-C., CHEN, B.-C., AND LIN, T.-W. 2010. Pats: A framework of pattern-aware trajectory search. In *Proceedings of the 1st Workshop on Uncertain Mobile Data Management and Mining (UMMM)*. 372–377.

YAN, Z., CHAKRABORTY, D., PARENT, C., SPACCAPIETRA, S., AND ABERER, K. 2011. Semitri: a framework for semantic annotation of heterogeneous trajectories. In *Proceedings of the 14th International Conference on Extending Database Technology (EDBT)*. 259–270.

YAN, Z., PARENT, C., SPACCAPIETRA, S., AND CHAKRABORTY, D. 2010. A hybrid model and computing platform for spatio-semantic trajectories. In *Proceedings of the 7th Extended Semantic Web Conference (ESWC)*. 60–75.

YAN, Z., SPREMIC, L., CHAKRABORTY, D., PARENT, C., SPACCAPIETRA, S., AND ABERER, K. 2010. Automatic construction and multi-level visualization of semantic trajectories. In *Proceedings of the 18th SIGSPATIAL International Conference on Advances in Geographic Information Systems (GIS)*. 524–525.

YOON, H., ZHENG, Y., XIE, X., AND WOO, W. 2009. Triptip: A trip planning service with tag-based recommendation. In *Proceedings of the 27th International Conference on Human Factors in Computing Systems (CHI)*. 3467–3472.

YOON, H., ZHENG, Y., XIE, X., AND WOO, W. 2010. Smart itinerary recommendation based on user- generated gps trajectories. In *Proceedings of the 7th International Conference on Ubiquitous Intelligence and Computing (UIC)*. 19–34.

YOON, H., ZHENG, Y., XIE, X., AND WOO, W. 2011. Social itinerary recommendation from user-generated digital trails. *Personal Ubiq. Comput. 16,* 5, 469–484.

ZHENG, V. W., ZHENG, Y., XIE, X., AND YANG, Q. 2009. Collaborative location and activity recommendations with gps history data. In *Proceedings of the 19th International Conference on World Wide Web (WWW)*. 1029–1038.

ZHENG, Y., LIU, L., WANG, L., AND XIE, X. 2008. Learning transportation mode from raw gps data for geographic applications on the web. In *Proceedings of the 17th International Conference on World Wide Web (WWW)*. 247–256.

ZHENG, Y. AND XIE, X. 2011. Learning travel recommendations from user-generated gps traces. *ACM Trans. Intell. Syst. Technol. 2,* 1, 2–19.

ZHENG, Y., ZHANG, L., XIE, X., AND MA, W.-Y. 2009. Mining interesting locations and travel sequences from gps trajectories for mobile users. In *Proceedings of the 18th International Conference on World Wide Web*. 791–800. ACM Press, New York.

Received June 2011; revised October 2011; accepted December 2011