Journal of Computer and System Sciences 82 (2016) 582–593

Contents lists available at ScienceDirect

Journal of Computer and System Sciences

www.elsevier.com/locate/jcss

Discovery of stop regions for understanding repeat travel behaviors of moving objects

Guangyan Huang [a,*], Jing He [b], Wanlei Zhou [a], Guang-Li Huang [d], Limin Guo [c], Xiangmin Zhou [d], Feiyi Tang [b]

[a] School of Information Technology, Deakin University, Melbourne, Australia
[b] College of Engineering and Science, Victoria University, Melbourne, Australia
[c] Institute of Software, Chinese Academy of Sciences, Beijing, China
[d] School of Computer Science and Information Technology, RMIT University, Melbourne, Australia

ARTICLE INFO

Article history:
Received 1 April 2015
Received in revised form 1 July 2015
Accepted 28 October 2015
Available online 3 December 2015

Keywords:
Trajectory mining
Repeat travel behaviors
Repeat route patterns
Stop regions

ABSTRACT

GPS trajectory dataset with high sampling-rates is usually in large volume that challenges the processing efficiency. Most of the data points on trajectories are useless. This paper summarizes trajectories using stop points. We define a new concept of stay stability (i.e., time dividing distance or reciprocal of speed) between any two GPS points to detect stop points on individual trajectories. We propose a novel Mining Repeat Travel Behaviors Using Stop Regions (MRTBUSR) method. In MRTBUSR, a stop region is a popular region containing a certain number of close stop points that can be grouped into a cluster. We then retrieve common sequences of stop regions to denote repeat route patterns and further analyze the stop durations on a stop region to find repeat travel behaviors. The experiments on 20 labeled trajectories selected from GeoLife demonstrated the semantic effect, accuracy and near linear efficiency of our proposed method.

1. Introduction

GPS trajectories have been widely used for travel experience sharing, life logging, sports activity analysis and multimedia content management, etc. [1]. However, it is not convenient to directly use raw GPS data (coordinates and timestamps) [1] and the raw data are hard to be understood by human, since raw data that are collected at high sampling-rates by machine usually are accumulated in large volume and comprise a great ratio of meaningless points. It is a challenge for handling big-sized data, and thus detecting key points to reduce the data size is an important task before mining trajectories.

In our previous work [2], we use a sequence of turning region IDs to approximate an original trajectory and greatly reduce the data size. Unfortunately, the turning regions that naturally simplify the trajectories are not equal to interesting regions. In contrast, stop regions, which are the places frequently visited by a certain number of users, such as culturally important places and commonly frequented public areas, are more meaningful to study the users' behaviors [1]. Simplifying trajectories using meaningful stop regions helps us to understand the behavior of moving objects. But stop points (i.e., instances of stop regions) are implicit on raw trajectories, which are generally displayed as the repeat similar locations.

* Corresponding author at: 221 Burwood highway, Burwood, VIC 3125, Australia.
E-mail addresses: guangyan.huang@gmail.com (G. Huang), Jing.He@vu.edu.au (J. He), Wanlei.Zhou@deakin.edu.au (W. Zhou), Guangli.Huang@qq.com (G.L. Huang), limin@nfs.iscas.ac.cn (L. Guo), xiangmin.zhou@rmit.edu.au (X. Zhou), feiyi.tang@live.vu.edu.au (F. Tang).

http://dx.doi.org/10.1016/j.jcss.2015.10.006
0022-0000/

In this paper, we efficiently and accurately detect stop points/regions, use stop regions with stop durations to simplify trajectories and discover repeat travel behaviors for trajectory understanding. We define a new concept of stay stability (i.e., time dividing distance or reciprocal of speed) between any two GPS points to detect stop points on individual trajectories, and propose a novel Mining Repeat Travel Behaviors Using Stop Regions (MRTBUSR) method. In MRTBUSR, a stop region is a popular region containing a certain number of close stop points that can be grouped into a cluster. We then retrieve common sequences of stop regions to denote repeat stay routes and further analyze the stop durations on a stop region to find repeat travel behaviors. The experiments on 20 labeled trajectories selected from GeoLife [1,3] demonstrated the semantic effect, accuracy and near linear efficiency of our proposed method. Particularly, using stop regions with stop durations to summarize trajectories in a semantic way enables easier understanding of behaviors of moving objects and allows manual analysis of massive trajectories.

The most related work is provided in [1] and [3], where a concept of stay points is very similar to our stop regions. A stay point is defined as a geographic region where a user stayed over a certain time interval [1]. Different from the definition of stay points, we do not use an absolutely interval threshold to define our stop regions; instead, a stop region is a special place where moving objects entering with relatively lower speed compared to their speeds before and after entering this place; we define a stay stability concept to measure this near-to-stop relatively low speed. Therefore, our stop regions are more flexible than stay points in [1] to capture more meaningful regions and are more effective to help discover repeat travel behaviors of moving objects.

Stop regions are vital to represent repeat travel behaviors. Three applications are

- *common trip routes*, denoted by sequences of stop regions of public transportations, such as train, which can be used to find return trips (the same sequence of key stop regions between two places);
- *popular visiting routes of special places*, such as beauty/historical spots, which can be recommended to others;
- *personal daily travel routes*, denoted by sequences of key stop regions for work, study and living, which can be used to study personal daily travel behaviors.

The rest of this paper is organized as follows. We present related work in Section 2 and define the problem in Section 3. The MRTBUSR method is proposed in Section 4, while its performance is evaluated in Section 5. Finally, Section 6 concludes the paper.

2. Related work

In this section, we conduct a survey of the state-of-the-art techniques about trajectory understanding for user behavior analysis [1,3], where similarity measurement of trajectories is critical. Particularly, we brief our previous work on both trajectory simplification [2] and using the duration of staying on each location of trajectories to discover common behaviors of objects [4].

GPS Trajectories record human behaviors. The same human behavior pattern cannot repeat exactly because of uncertainties introduced by discrete sampling and sampling error [5], as well as individual difference of moving objects. So, similarity measurement of trajectories is a basis for trajectory mining.

The simplest similarity measurement only considers spatial difference of trajectories, such as Euclidean-based distance measures. It is more precise to measure both the spatial (i.e., locations) and temporal (i.e., timestamps) similarities of trajectories. Dynamic Time Wrapping (DTW) [6] allows similar shapes to match even when they are in different time phase. Both Edit Distance on Real sequence (EDR) [7] and Longest Common Subsequences (LCSS) [8] are used to measure the similarity between trajectories that contain noise and local time shifting. EDR is more robust than Euclidean distance and DTW and more accurate than LCSS. TRACLUS [9] is used to discover common sub-trajectories.

To further understand user behaviors, extracting behavior knowledge in semantic aspects other than in merely spatial temporal data is more effective. For example, the similarity of movement patterns are extracted from trajectories [11,12], and also some special behavior phenomena, like the silent durations, are considered for clue-aware trajectory similarity measurement in [13]. Here, the semantic trajectory [14–17] means a set of spatial temporal positions complemented with semantic annotations. Semantic annotations can be a label of geographic information, such as the real-world address corresponding to a longitude and latitude location, or a label of a traveling mode, such as by train, walk and driving car. In [18], the similarity between two users in terms of the similarity of their maximal semantic trajectory patterns are considered. A semantic hierarchical tree-structured framework [1,19,20] is provided to calculate similarity, and both spatial and semantic feature are considered for trajectory analysis [21,22].

Other work considers context in addition to trajectories; for example, in [10], both moving path and user groups in Location Based Service environments are used to predict the next behavior of mobile users.

In [2], we proposed a novel mining algorithm for Longest Common Route (LCR) patterns based on turning regions (LCR-Turning), which discovers a sequence of turning regions to abstract a trajectory; this method keeps the coarse shapes of trajectories. But we have observed that many turning regions are meaningless; they are not interesting landmark regions. Instead, the stop regions discovered by this paper, where people stay for a longer time, are more meaningful.

Different from the above existing work, we argue that the stop points with staying durations of a trajectory are the most important clues for analysis of human behaviors. At a stop point, user's location is in a bounded region/area and how long

the user stays on the stop point reflect the meaning/purpose of his/her stay behavior. Also, analysis of staying patterns of users need not semantic annotations, instead, it can help automatically generate semantic annotations.

So, in our recent work [4], we developed a novel approach for discovering common behaviors by considering the duration of staying on each location of trajectories (DoSTra). The duration of staying on a location is important. DoSTra can easily to differentiate user behaviors by considering duration of staying on each location when the users follow the same route (a sequence of regions) and thus help detect the group that has similar lifestyle, habit or behavior patterns.

The most related work is provided in [1] and [3], where a concept of stay point, is very similar to our stop regions. A stay point is defined as a geographic region where a user stayed over a certain time interval [1]. Similar strategies are used to detect arbitrary shape of the interesting regions: OPITCS is used for stay points in [1] and DBSCAN [23] is used for stop regions in this paper. But different from the definition of stay points, we do not use an absolutely interval threshold to define our stop regions; instead, a stop region is a special place where moving objects entering with relatively lower speed compared to their speeds before and after entering this place. Thus, our stop regions are more flexible than stay points in [1] to capture more meaningful regions and are more effective to help discover repeat travel behaviors of moving objects.

In this paper, we extend further the work in [4] and provide a more flexible and systematic solution to discover repeat travel routes denoted by sequences of stop regions from GPS trajectories to understand human behaviors.

3. Problem definition

3.1. Basic terms

We formally define basic terms and then model the problem.

Definition 1 *(GPS trajectory).* (See [1].) A GPS trajectory is a sequence of points, $traj = < p_1, \ldots, p_u, \ldots, p_k >$, where $p_u = (lat_u, lgt_u, t_u)$, t_u $(u = 0..k)$ is a timestamp (yy–mm–dd, hh–mm–ss) for a snapshot, $\forall_{0 \le u < k}$, $t_u < t_{u+1}$, and (lat_u, lgt_u) are 2-D locations denoted by latitude and longitude.

Definition 2 *(Stay stability).* The stay stability on $traj_{BA}$ (a trajectory segment between locations: A and B) is denoted by

$$S_{stay}(A, B) = \frac{dT}{len(traj_{BA})}, \tag{1}$$

where $len(traj_{BA})$ is the length of $traj_{BA}$, $dT = T_B - T_A$ is the duration on $traj_{BA}$, and T_A and T_B are timestamps on locations A and B, respectively.

According to Definition 2, the stay stability is the reciprocal of the speed on a trajectory segment, the slower the velocity, the greater the stay stability.

Definition 3 *(Stop points).* A point between A and B is a stop point, if its stay stability is greater than a threshold, N_{coef}, times of the average stay stability on $traj_{BA}$, $S_{stay}(A, B)$.

Definition 4 *(Stop regions).* If a group of stop points from more than *min_sup* trajectories are clustered into a cluster, we call this stop point cluster a stop region. Stop points are instances of a stop region.

Definition 5 *(Stop duration).* is the duration of staying at a stop region.

After having studied common behaviors of objects using the duration of staying on each location of trajectories in [4], we define a more complete and formal concept — repeat travel behaviors in Definition 6.

Definition 6 *(Repeat travel behaviors (or repeat stay patterns)).* A *repeat stay route* is a sequence of at least k stop regions and the whole route repeats at least l times. A *repeat travel behavior* is a *repeat stay pattern*, where each stop region with similar (with a bounded error of ε) stop duration on a *repeat stay route*.

So, we group the three typical applications mentioned in Section 1 into two classes:

- *repeat stay routes*: *common trip routes* of multiple moving objects or *return trips* of one moving object and *popular visiting routes of special places*;
- *repeat stay patterns*: *personal daily travel patterns*.

Table 1
The MRTBUSR method.

Input: N_{coef}, ***min_sup***, k, l and ε.	
Output: Repeat travel behaviors.	
Step 1.	**Detect stop points (Algorithm 1).** Compute stay stabilities (see Definition 2). Find *stop points* with *stay stability* greater than N_{coef} times of the average stay stability of the whole trajectory (see Definition 3).
Step 2.	**Label trajectories using stop regions (Algorithm 2).** Group stop points into clusters to identify *stop regions* with at least ***min_sup*** supports (see Definition 4). Merge the same consecutive stop regions on a trajectory and compute *stop duration* (see Definition 5) of this stop region. Use a sequence of stop region IDs with stop duration to label a trajectory.
Step 3.	**Find repeat travel behaviors (Algorithm 3).** Discover *repeat travel behaviors* of moving objects with at least a sequence of k *stop regions* and the whole sequence repeats at least l times with a bounded *stay duration* difference, ε (see Definition 6).

Table 2
An example.

Point	(Latitude, longitude)	Time	Stay stability
A	(41.688133, 82.828615)	39541.4726 (2008/4/3 11:20:33)	0.0389
B	(41.695882, 82.847058)	39541.47329 (2008/4/3 11:21:32)	

3.2. Overview of the proposed method

We propose a novel Mining Repeat Travel Behaviors Using Stop Regions (MRTBUSR) method as shown in Table 1 and develop three algorithms to implement the MRTBUSR method in three steps, respectively. In *Step 1*, we detect stop points (see Algorithm 1 in Section 4.2). In *Step 2*, we label trajectories using stop regions (see Algorithm 2 in Section 4.3). In *Step 3*, we find repeat travel behaviors (see Algorithm 3 in Section 4.4).

4. Mining Repeat Travel Behaviors Using Stop Regions

In this section, we present the details of the three algorithms in the MRTBUSR method.

4.1. Computing the stay stabilities

We calculate the distance between two locations denoted by latitude and longitude by

$$len(traj_{BA}) = 2\alpha R \times arcsin\sqrt{sin^2(\frac{lat_B - lat_A}{2}) + cos(lat_A) \times cos(lat_B) \times sin^2(\frac{lgt_B - lgt_A}{2})} \quad (2)$$

where the radius of the Earth is $R = 6\,367\,000$ meters and the coefficient is set to $\alpha = 10^{-5}$ for simplifying the value of a stay stability. We explain how to calculate the stay stabilities using an example of two points in Table 2. In Table 2, $lat_A = 41.688133$, $lgt_A = 82.828615$, $lat_B = 41.695882$, $lgt_B = 82.847058$, $T_A = 39541.4726$ and $T_B = 39541.47329$, and according to Equations (1) and (2), $S_{stay}(A, B) = 0.0389$.

4.2. Detecting stop points

We first use examples to explain how to detect stop points by one cut under both single and multiple transportation modes, and then provide a general OneCut algorithm.

4.2.1. Trajectories under single transportation modes

In Fig. 1, we use the average stay stabilities of the whole curve to detect stop points for single transportation modes; that is, $N_{coef} = 1$. The maximum value of stay stability is set to around $\tau = 5$ times of the cut point, since the stay stability approximates to infinite when the distance is zero. Fig. 1 shows the effectiveness of the cut on the stay stability curves. The points above the cut can be used to compute stop points and the points below the cut can be neglected, particularly for the curves of train and taxi. The cut for the walk curve generates a large number of stop points; this captures an interesting human behavior patterns that people frequently stop when walking and can help automatically separate walking segment of trajectories from train/taxi segments of trajectories.

4.2.2. Trajectories under multiple transportation modes

In Fig. 2, we separate different transportation modes of a trajectory using a cut on the stay stability curve of this trajectory.

Fig. 2 (a) plots an example of an original GPS trajectory that comprises four partitions in three transportation modes: by train, walking and by taxi. Fig. 2 (b) accordingly shows the stay stability curve of the trajectory in Fig. 2 (a). We can

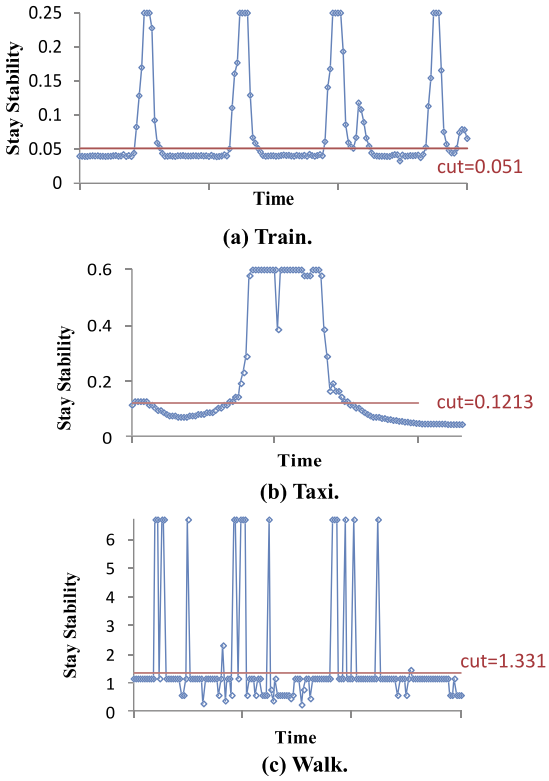


Fig. 1. Detecting stop points under single transportation modes ($N_{coef} = 1$).

see from Fig. 2 (b) that the stay stabilities by train are close to zero, the stay stabilities by taxi are between 0 and 0.5, and the stay stabilities by walking are greater than 0.5, actually, most are greater than 1. That is, the stay stabilities during walking are far greater than those under other transportation modes. Also, the values of stay stabilities vibrate rapidly when switching transportation modes. Using stay stabilities can help automatically partition trajectories into different modes, just like the work done in [3] to segment walk modes. Those intervals with great stay stabilities such as the peak in the curve of Fig. 2 (b) help to identify stop points, so we identify stop points using a cut (i.e., the stay stability of the whole segment). The points above the cut are used to compute stop points. The maximum value of stay stability is set to $\tau = 10$ times of the cut point for Fig. 2 (b). Here, the setting of the value of τ is only for display purpose.

4.2.3. The OneCut algorithm for detecting stop points

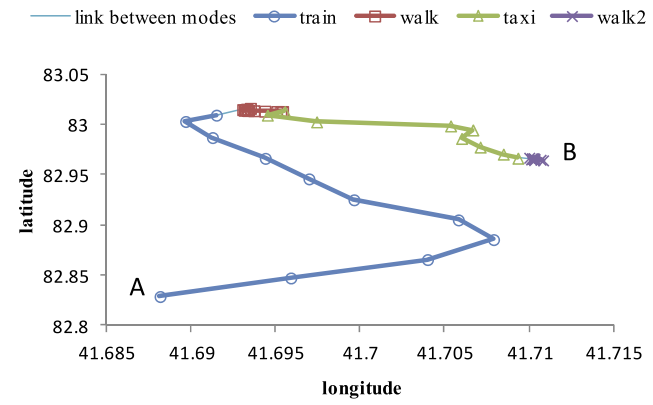
According to Equations (1) and (2), we compute the stay stability of the trajectory, $traj_{BA}$, in Fig. 2 (a) below:

$$S_{stay}(A, B) = \frac{dT}{len(traj_{BA})} = \frac{\Sigma_{i=1}^{n}(\Delta t_i)}{\Sigma_{i=1}^{n}(\Delta d_i)}, \tag{3}$$

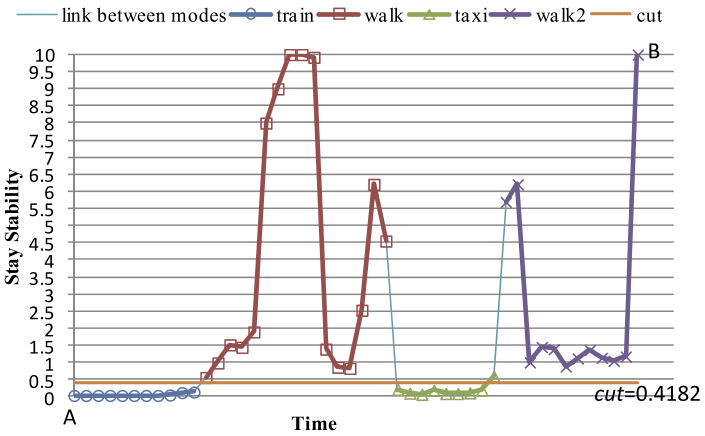
where $\Delta t_i = t_{i+1} - t_i$ is the interval between two consecutive points: p_{i+1} and p_i ($p_i = (lat_i, lgt_i, t_i)$), $\Delta d_i = len(traj_{p_{i+1}p_i})$ and n is the number of intervals between A and B in Fig. 2 (a).

We use the example in Fig. 2 to explain Algorithm 1 (in Table 3). In *Step 1*, we generate a curve of stay stabilities of a moving object. In *Step 2*, we use Eq. (3) to compute the stay stability of the whole trajectory, $S_{stay}(A, B) = 0.1394$. In *Step 3*, we set $N_{coef} = 3$ and get $cut = N_{coef} \times S_{stay}(A, B) = 0.4182$; the cut coarsely partitions the trajectory into four segments as shown in Fig. 2 (b). In Step 4, we simplify a trajectory using a sequence of stop points denoted by a 2-tuple (average location, stop duration). The OneCut algorithm can use to help detect walk segments in [3], where walk segments are used to separate different transportation mode pieces.

The stop points detected by our OneCut algorithm are meaningful to analyze human behavior and an example in Fig. 3 shows the effectiveness. Fig. 3 shows a stop region map, which is generated using our OneCut algorithm from a GPS user's trajectory ("010" in GeoLife) collected for over 1.5 years and displays the user's behavior and life style. We have marked three different types of stop points (i.e., stay 5+ minutes, stay 30+ minutes and stay 2+ hours) on the map, which help understand the user's behavior in over 1.5 years.



(a) An Original GPS Trajectory.



(b) A Stay Stability Curve with A Cut=0.4182 (N_{coef}=3).

Fig. 2. Analysis of stay stabilities on a trajectory under various travel modes.

Table 3
The OneCut algorithm.

Algorithm 1. Detecting stop points using one cut.	
Input: A GPS trajectory, $traj_{BA}$.	
Output: A sequence of stop points on $traj_{BA}$.	
Step 1.	Calculate stay stabilities on each interval of a trajectory to form a curve, $\mathbb{S}$.
Step 2.	Compute the stay stability of the whole trajectory, $S_{stay}(A, B)$, using Eq. (3).
Step 3.	Use $cut = N_{coef} * S_{stay}(A, B)$ to cut the trajectory curve $\mathbb{S}$ into m partitions with two modes: stable (partitions above the cut line) and non-stable (partitions below the cut line), where N_{coef} is a coefficient.
Step 4.	For each partition with mode equal to "stable", Compute a stop point denoted by an average location (latitude, longitude) and a duration (i.e. the end time subtract the start time); Output a sequence of stop points on $traj_{BA}$ in time order.

4.3. The algorithm of labeling trajectories using stop regions

We present the main idea of the Trajectory Labeling (TL) algorithm, as shown in Algorithm 2 (Table 4), as follows. We use DBSCAN [23] to cluster a group of close stop points (that are identified by the OneCut algorithm) into a stop region (*Step 1*) and a stop point is an instance of a stop region. Then we identify stop regions (*Step 2*) and compute the location of stop regions (*Step 3*). Finally, we use a sequence of stop region IDs to label a trajectory (*Step 4*). Algorithm 2 changes each original trajectory into sequences of stop region IDs with stop duration as shown in Fig. 4; this is easier to be understood by people. Note that not every stop point can be grouped into a stop region; for example, those with label "00" as shown in Fig. 4 are neglected, since they are not repeated at a minimal number of times required for a stop region.

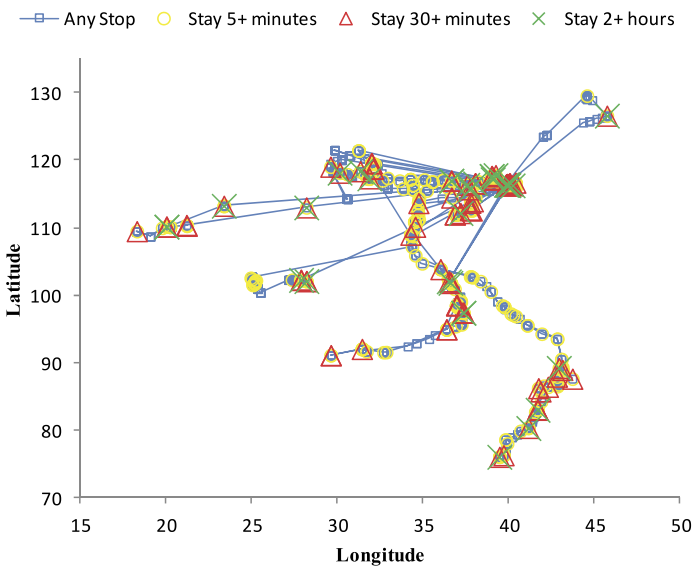


Fig. 3. Stop region map of a GPS user ("010" in GeoLife) for over 1.5 years.

Table 4
The Trajectory Labeling algorithm.

Algorithm 2. Labeling trajectories using stop regions.

Input: A trajectory, $traj_{raw} =< subtraj_1, subtraj_2, \ldots, subtraj_i, \ldots, subtraj_m >$, where $subtraj_i =< p_{1i}, \ldots, p_{ui}, \ldots, p_{ki} >$, where $p_{ui} = (lat_{ui}, lngt_{ui}, t_{ui})$, and a stop point set on $traj_{raw}$, $Set_{stop} = \{q_1, \ldots, q_j, \ldots, q_x\}$, where $q_j = (lat_j, lgt_j, t_j, subtrajID_j)$;

Output: A labeled trajectory with m sequences of stop regions, $traj_{label} =< lsubtraj_1, lsubtraj_2, \ldots, lsubtraj_i, \ldots, lsubtraj_m >$, where $lsubtraj_i =< r_{1i}, \ldots, r_{vi}, \ldots, r_{ki} >$, $r_{vi} = (lat_{vi}, lgt_{vi}, StopID_{vi}, t1_{vi}, t2_{vi})$, the center location of stop region, $StopID_{vi}$, is (lat_{vi}, lgt_{vi}), $t1_{vi}$ is the start time and $t2_{vi}$ is the end time for entering stop region, $StopID_{vi}$.

Step 1. Use the DBSCAN algorithm to cluster stop points in Set_{stop} by locations, with parameters of *eps* and *Minpts*;

Step 2. Identify stop point clusters (stop regions) from the DBSCAN clusters that satisfy at least k different sub-trajectory IDs;

Step 3. Denote a stop point cluster (stop region) by $(lat_c, lgt_c, StopID)$, where (lat_c, lgt_c) is the center location of all stop points in this cluster;

Step 4. Change each raw trajectory by sequences of labels. A label of the ith stop region on the vth sub-trajectory is denoted by $r_{vi} = (lat_{vi}, lgt_{vi}, StopID_{vi}, t1_{vi}, t2_{vi})$. A consecutive sequence of stop points within the same stop region is merged as one.

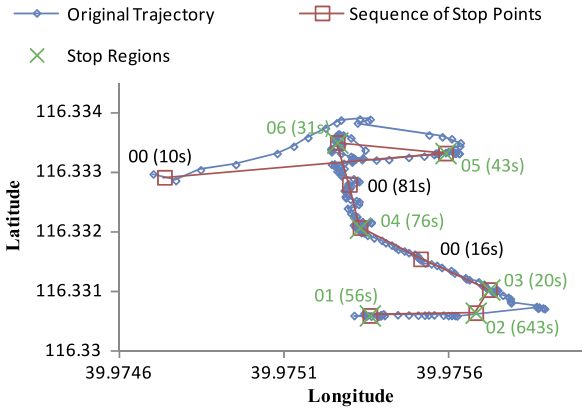


Fig. 4. An example of labeled trajectory using stop regions ("01, 02, 03, 04, 05, 06"). The original trajectory piece is from GeoLife, 073 dataset, 2008/5/23, 9:58:39–10:17:27.

4.4. Mining repeat travel behaviors of moving objects

A repeat travel behavior occurs on a repeat route. If a certain number of objects also stay at each stop region within similar durations, we call it a repeat stay pattern. So, we first retrieve repeat stay routes in Section 4.4.1 and then find repeat stay patterns further in Section 4.4.2.

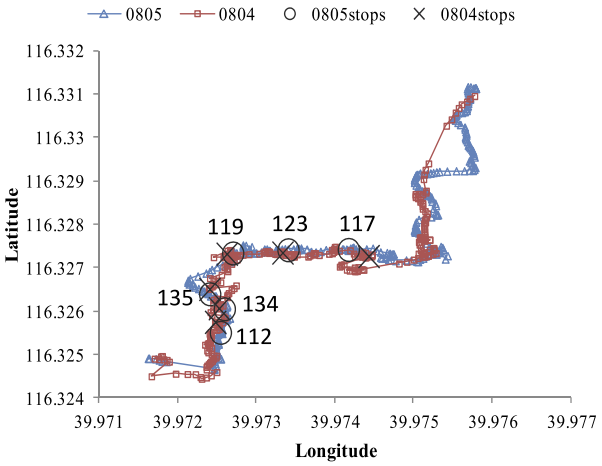


Fig. 5. An example repeat stay pattern detected from Geolife, 073 dataset, 0805 (2008/8/5 2:16:22–2:29:14) and 0804 (2008/8/4 2:14:27–2:28:15).

4.4.1. Finding repeat stay routes

We assume that no stop region is missed on a route since high sampling-rates ensure the precision of the original trajectories. That is, "$A->B->C->D$" and "$A->B->D$" are definitely two different routes from A to D. We use suffix trees to efficiently retrieve common sequences of stop region IDs for representing repeat stay routes. We use words (i.e., a sequence of letters/numbers), instead of letters, to denote a stop region ID and thus the number of stop regions is unlimited. For example, we use words of meaningful location names to represent locations A, B, C and D, and denote the route by "*flinders, parliament, flagstaff, newmarket*", where one word denotes one location name and a comma (",") or a blank space is used to separate two consecutive location names.

We introduce the basic concept of suffix trees. A suffix tree with a word-based alphabet is as follows. $Str =< W_1, W_2, \ldots, W_i, \ldots, W_n >$ is a string, where W_i is the ith word in Str, then $Suf_i =< W_i, W_{i+1}, \ldots, W_n >$ is the ith suffix of Str that starts at the ith word. The suffix tree for the string Str of length n is defined as a tree such that (in [24]):

- The paths from the root to the leaves have a one-to-one relationship with the suffixes of Str,
- Edges spell non-empty strings, and
- All internal nodes (except the root) have at least two children.

For example, a string of 3 stop regions, "*flinders parliament flagstaff*", has three suffixes: "*flinders parliament flagstaff*", "*parliament flagstaff*" and "*flagstaff*".

We provide a special suffix tree structure which can store the support list of sub-trajectory IDs for each common sub-sequences. We use a hash table to store the number of supports (*nSupports*) and the support list, and thus we can easily achieve common sub-sequences (i.e., repeat stay routes) that have at least k words (i.e., stop regions) and $nSupports \geq l$ supports (i.e., sub-trajectory IDs). The parameters k and l are defined in Definition 6. Note that we consider the return route as the same route; for example, "*flinders parliament flagstaff*" and "*flagstaff parliament flinders*" are the same route. Fig. 5 shows an example of a repeat stay route.

4.4.2. Discovering repeat stay patterns

We propose Algorithm 3 in Table 5 to discover repeat stay patterns by clustering instances based on stop durations on a repeat stay route. After labeling trajectories by Algorithm 2 (in Table 4), instances (i.e., sub-trajectories) that support a repeat stay route are sequences of labels. A label of the ith stop region on the vth sub-trajectory is denoted by $r_{vi} = (lat_i, lgt_i, StopID_i, t1_{vi}, t2_{vi})$, where $StopID_{vi}$ is an ID of the stop region, the center location of stop region i is (lat_i, lgt_i), $t1_{vi}$ and $t2_{vi}$ are the start and the end timestamps for entering the stop region. So, $R_i = (lat_i, lgt_i, StopID_i)$ and $\Delta T_{vi} = |t2_{vi} - t1_{vi}|$.

In Algorithm 3, we group instances (i.e., curves of stop durations) that support a Repeat Stay Route into different clusters; all instances (no less than *min_instances*) in each cluster have a sequence of at least, *min_durations*, consecutive similar stop durations. In Algorithm 3, we use our previously developed Clustering Points On a Line (CPOL) algorithm [25] to cluster a 1-dimension dataset at *Line 2*. The main idea of the procedure CPOL, (Q, E_{max}, N_{min}), is as follows. Q denotes a set of 1-dimension data points that are sorted along the X-coordinator. E_{max} is the maximum bounded error to the center of a cluster and N_{min} is the minimal number of points required to form a cluster. First, the center of all the data points in Q is computed as c_0. All data points in the neighborhood of c_0, a circle area centered by c_0 with a radius of E_{max}, are grouped into a cluster, C_1, which naturally separates the remaining data points into Q_{left} and Q_{right}. Then, in the same way, we recursively group data points in Q_{left} (or Q_{right}) into clusters. It is possible Q_{left} (or Q_{right}) $= \Phi$ in any recursive step.

Table 5
Algorithm of discovering repeat stay patterns.

Algorithm 3. Discovering repeat stay patterns (*min_instances*, *min_durations*).

Input: A set of m instances of a repeat stay route, $\mathbb{R} = \{I_1, \ldots, I_j, \ldots, I_m\}$, where the jth instance is denoted by $I_j =< (R_1, \Delta T_{1j}), \ldots, (R_i, \Delta T_{ij}), \ldots, (R_k, \Delta T_{kj}) >$ and the sequence of k region IDs of each instance for this repeat stay route is the same: $R =< R_1, \ldots, R_i, \ldots, R_k >$.

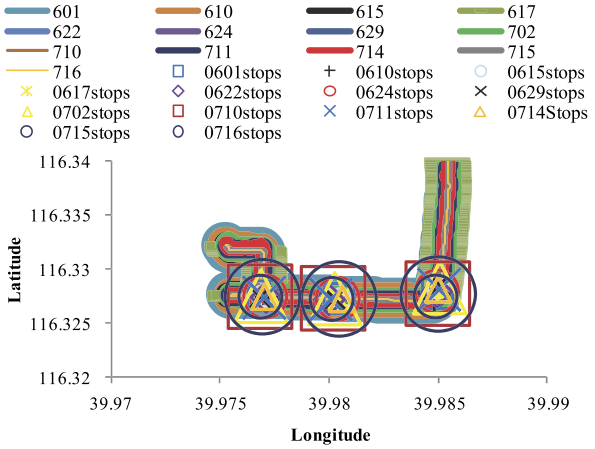
Output: $\mathbb{N}$: a set of repeat stay patterns.

1 Let $Q =< Q_1, \ldots, Q_i, \ldots, Q_k >$, where $Q_1 = \{\Delta T_{11}, \ldots, \Delta T_{1j}, \ldots, \Delta T_{1m}\}$ and $\Delta T_{11} \le \Delta T_{1j} \le \Delta T_{1m}$

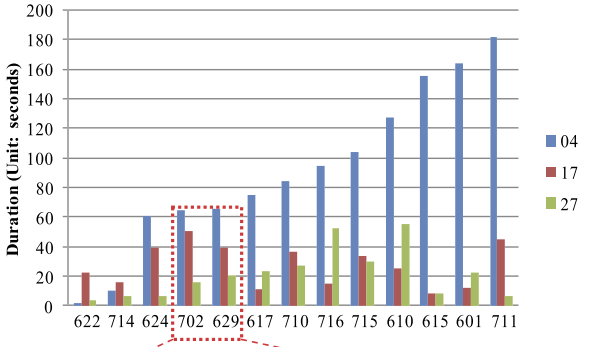
2 $\mathbb{C}_i = \textbf{CPOL}(Q_i, \gamma, N_{min})$;// $\mathbb{C}_i$ is a set of clusters of similar durations at Region i.

3 If ($\mathbb{C}_i \neq \emptyset$ and $(i+1) \le k$)
For each cluster in $\mathbb{C}_i$
{i++;
Go to Line 2;// to check the similarity of durations at the next Region, i, for the data points in current $\mathbb{C}_{i-1}$.}

4 **Output** those clusters of instances, as $\mathbb{N}$ (a set of repeat stay patterns). Each cluster is a repeat stay pattern, where the minimal number of instances is *min_instances* and all the instances have a sequence of at least *min_durations* $\le k$ similar durations (within a bounded time error, γ).



(a) Spatial Pattern (Repeat Sequence of Stop Regions).

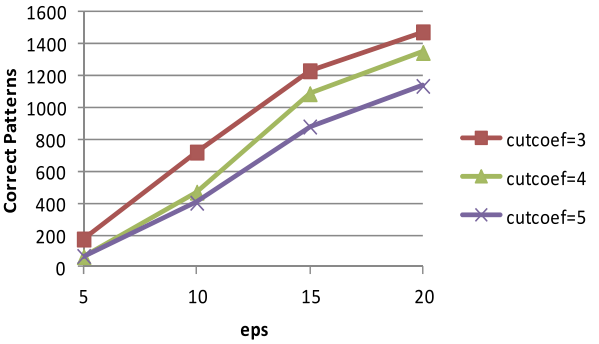


Trajectory ID	Stop 04	Stop 17	Stop 27
702 (2008/7/2)	64.34	51.06	16.34
629 (2008/6/29)	65.36	39.83	20.42

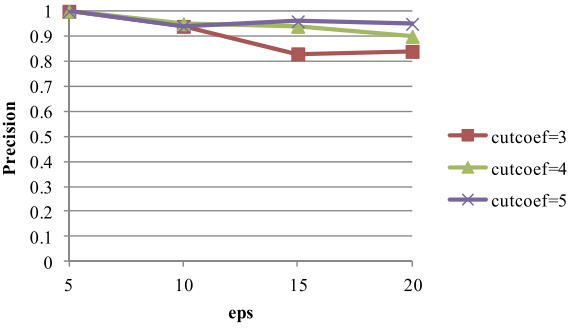
(b) Temporal Patterns (Similar Stay Durations on Three Stop Regions), γ=15.

Fig. 6. A stay pattern example on a segment of trajectory 115 in GeoLife, from 2008/6/1 (601) to 2008/7/16 (716). (For interpretation of the references to color in this figure, the reader is referred to the web version of this article.)

We use examples in Fig. 6 to explain Algorithm 3. We obtain repeat sequences of stop regions by Algorithm 2 as shown in Fig. 6 (a), where the repeat pattern has three stop regions, supported by 13 instances (601, 610, 615, 617, 622, 624, 629, 702, 710, 711, 714, 715, 716). Fig. 6 (b) shows an example of the output of Algorithm 3 by checking the similarity of stop



(a) Weighted Amount of Correct Patterns.



(b) Precision.

Fig. 7. *cutcoef* With $d_{error} = 50$ m.

durations, where "702" and "629" show a repeat travel behavior since both of them stay at each of the three stops 04, 17 and 27 with very similar stop durations as shown in the table in Fig. 6 (b), given a bounded time error, $\gamma = 15$ seconds.

5. Performance evaluation

In this section, we evaluate the effectiveness (precisions and number of patterns retrieved) and efficiency of our MRTBUSR method using 20 labeled GPS trajectories selected from GeoLife. We also discuss the optimal parameters of MRTBUSR. Note that the examples that show the semantic effect of retrieved patterns can be found in Figs. 1–6.

5.1. Dataset and experimental setup

We have selected 20 labeled GPS trajectory datasets from GeoLife [1,3,26] for our experimental study; each comprises 110+ trajectories and total of 9462 trajectories with data size of 688 MB. We set $MaxStability = 10$. To obtain the maximum number of stops, we set $N_{coef} = 3$, $minDuration = 0$. If the stop points on an individual trajectory are in stop regions, they can be discovered by Algorithm 2; otherwise, they are neglected.

$$X = \frac{1}{n} \times \Sigma_{i=1}^{n}(x_i - \bar{x})^2, \tag{4}$$

where $\bar{x} = \frac{1}{n} \times \Sigma_{i=1}^{n}(x_i)$.

We use variance to compute the distance error of each stop region instance on a pattern, as shown in Eq. (4). We set a bounded error threshold, d_{error}, to evaluate the precision; that is, in a discovered repeat travel pattern, we check the correctness of a region ID by considering if all the instances for this region ID are within the bounded d_{error} meters. A small variance indicates that the data points tend to be very close to the mean.

All the experiments have been conducted on a workstation [Intel(R) Xeon (R) CPU E5-2620v3 (12 cores: 6 cores 2 processors) @ 2.40 GHz, 32 GB of RAM, Windows 8.1]. The suffix tree for retrieving common sub-sequences are implemented in Eclipse (Java) and other algorithms are implemented in PHP.

5.2. Study of optimal parameters

We set $minPts = 2$. The other two parameters: *eps* and d_{error} are determined by applications. Generally, we set $d_{error} = 5 \times eps$. So, we set $eps = 10$ m and $d_{error} = 50$ m. We analyze the optimal *cutcoef* in Fig. 7. We find *cutcoef* should be

Table 6
Discovered patterns from 20 trajectory datasets of 20 moving objects.

MO ID	010	020	062	065	067	068	073	084	085	092
$N_{patterns}$	77	130	524	53	158	1244	70	971	2419	59
Precision	1	0.94	0.91	0.88	0.9	0.73	1	0.96	0.99	0.99

MO ID	096	104	112	115	126	128	144	153	163	167
$N_{patterns}$	221	10	102	260	426	1704	263	1979	574	663
Precision	1	1	0.8	1	0.9	0.85	1	0.78	0.92	0.9

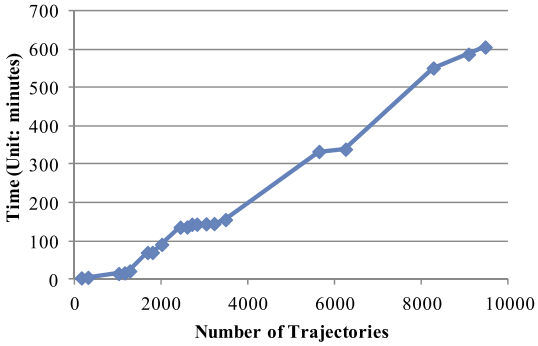


Fig. 8. Near linear run time.

greater than 2, just as shown in Figs. 1 and 2. We study the optimal value of *cutcoef* by analysis of their correct patterns and precisions as shown in Fig. 7. We choose $cutcoef = 3$ as the optimal value, since we expect to retrieve more correct patterns while keeping the precision acceptable (greater than 80%).

5.3. Effectiveness

After studying the optimal parameters, we set $minPts = 2$, $eps = 10$ m, $cutcoef = 3$, $d_{error} = 50$ m and conduct experiments on 20 datasets.

The results are shown in Table 6, comprising both the number of repeat stay patterns discovered ($N_{patterns}$) and their precisions. We can see from Table 6 that the precision is lower when $N_{patterns}$ is greater, such as 068 ($precision = 0.73$ and $N_{patterns} = 1244$) and 153 ($precision = 0.78$ and $N_{patterns} = 1979$). Some examples of the detailed patterns are shown in Figs. 4, 5 and 6.

5.4. Efficiency

We also evaluate the efficiency of our method using the run time. We neglect the runtime of Algorithms 1 and 3, since the major time (over 98% of the total time) is consumed by running Algorithm 2. We can see from Fig. 8, our method shows a near linear run time efficiency, changing with the number of trajectories. This demonstrates that our method is efficient to process massive trajectory data.

6. Conclusions

In this paper, we have defined a new concept of stay stability (i.e., time dividing distance or reciprocal of speed) between any two GPS points and proposed a novel MRTBUSR method. In MRTBUSR, we detect stop points using a cut on a stay stability curve of an individual trajectory, cluster stop points into stop regions, retrieve common sequences of stop regions to denote common route patterns and further analyze stop durations on each stop regions to find repeat travel behaviors. The experiments on 20 labeled trajectories selected from GeoLife have demonstrated the semantic effect, accuracy and near linear efficiency of our MRTBUSR method.

Acknowledgments

This work was partially supported by Australian Research Council Grant (No. DE140100387) and the National Natural Science Foundation of China Grant (No. 61402449).

References

[1] Y. Zheng, L. Zhang, X. Xie, W.-Y. Ma, Mining interesting locations and travel sequences from GPS trajectories, in: Proceedings of International Conference on World Wild Web, WWW 2009, Madrid Spain, ACM Press, 2009, pp. 791–800.
[2] G. Huang, Y. Zhang, J. He, Z. Ding, Efficiently retrieving longest common route patterns of moving objects by summarizing turning regions, in: The 15th Pacific-Asia Conference on Knowledge Discovery and Data Mining, PAKDD 2011, 2011, pp. 375–386.
[3] Y. Zheng, Q. Li, Y. Chen, X. Xie, W.-Y. Ma, Understanding mobility based on GPS data, in: Proceedings of ACM Conference on Ubiquitous Computing, UbiComp'2008, Seoul, Korea, ACM Press, 2008, pp. 312–321.
[4] L. Guo, G. Huang, X. Gao, J. He, DoSTra: discovering common behaviors of objects using the duration of staying on each location of trajectories, in: AAAI'2015 Workshop, Austin, Texas, USA, 25–29 January 2015.
[5] J. He, Y. Zhang, G. Huang, P. de Souza, CIRCE: correcting imprecise readings and compressing excrescent points for querying common patterns in uncertain sensor streams, Inf. Syst. 38 (8) (2013) 1234–1251.
[6] E. Keogh, Exact indexing of dynamic time warping, in: VLDB'2002, 2002, pp. 406–417.
[7] L. Chen, M. Ozsu, V. Oria, Robust and fast similarity search for moving object trajectories, in: SIGMOD'2005, 2005, pp. 491–502.
[8] M. Vlachos, M. Hadjieleftheriou, D. Gunopulos, E.J. Keogh, Indexing multidimensional time-series, VLDB J. 15 (1) (2006) 1–20.
[9] J.-G. Lee, J. Han, K.-Y. Whang, Trajectory clustering: a partition-and-group framework, in: SIGMOD'2007, 2007, pp. 593–604.
[10] E.H.-C. Lu, V.S. Tseng, Mining cluster-based mobile sequential patterns in location-based service environments, in: MDM'2009, 2009, pp. 273–278.
[11] Q. Li, Y. Zheng, X. Xie, Y. Chen, W. Liu, W.-Y. Ma, Mining user similarity based on location history, in: ACM GIS'08, 2008.
[12] Y. Zheng, L. Zhang, Z. Ma, X. Xie, W.-Y. Ma, Recommending friends and locations based on individual location history, ACM Trans. Web 5 (1) (2011) 5.
[13] C.-C. Hung, W.-C. Peng, W.-C. Lee, Clustering and aggregating clues of trajectories for mining trajectory patterns and routes, VLDB J 24 (2) (2015) 169–192.
[14] L.O. Alvares, V. Bogorny, B. Kuijpers, B. Moelans, J.A.F. de Macedo, A.T. Palma, Towards semantic trajectory knowledge discovery, in: Data Mining and Knowledge Discovery, 2007.
[15] C. Parent, S. Spaccapietra, C. Renso, G. Andrienko, N. Andrienko, V. Bogorny, M.L. Damiani, A. Ckoulalas-Divanis, J. Macedo, N. Pelekis, Y. Theodoridis, Z. Yan, Semantic trajectories modeling and analysis, ACM Comput. Surv. 45 (4) (2013).
[16] S. Spaccapietra, C. Parent, Adding meaning to your steps, in: ER'2011, 2011, pp. 13–31.
[17] Z. Yan, D. Chakraborty, C. Parent, S. Spaccapietra, K. Aberer, SeMiTri: a framework for semantic annotation of heterogeneous trajectories, in: EDBT'2011, 2011, pp. 259–270.
[18] J.J.-C. Ying, E.H.-C. Lu, W.-C. Lee, T.-C. Weng, V.S. Tseng, Mining user similarity from semantic trajectories, in: ACM LBSN'2010, 2010, pp. 19–26.
[19] Y. Zheng, X. Xie, Learning travel recommendations from user-generated GPS traces, ACM Trans. Intell. Syst. Technol. 2 (1) (2011).
[20] Y. Zheng, L. Zhang, Z. Ma, X. Xie, W.-Y. Ma, Recommending friends and locations based on individual location history, ACM Trans. Web 5 (1) (2011).
[21] J.-C. Ying, H.-S. Chen, K.W. Lin, E.H.-C. Lu, V.S. Tseng, H.-W. Tsai, K.H. Cheng, S.-C. Lin, Semantic trajectory-based high utility item recommendation system, Expert Syst. Appl. (2014) 4762–4776.
[22] H. Liu, M. Schneider, Similarity measurement of moving object trajectories, in: IWGS'2012, 2012, pp. 19–22.
[23] M. Ester, H.-P. Kriegel, J. Sander, X. Xu, A density-based algorithm for discovering clusters in large spatial databases with noise, in: SIGKDD'1996, 1996, pp. 226–231.
[24] G. Dan, Algorithms on Strings, Trees and Sequences: Computer Science and Computational Biology, Cambridge University Press, USA, 1997.
[25] G. Huang, Y. Zhang, J. Cao, M. Steyn, K. Taraporewalla, Online mining abnormal period patterns from multiple medical sensor data streams, World Wide Web 17 (4) (2014) 569–587.
[26] Y. Zheng, X. Xie, W.-Y. Ma, GeoLife: a collaborative social networking service among user, location and trajectory, IEEE Data Eng. Bull. 33 (2) (2010) 32–40.