DoSTra: Discovering Common Behaviors of Objects Using the Duration of Staying on Each Location of Trajectories

Limin Guo[1], Guangyan Huang[2], Xu Gao[1], Jing He[3,4], Bin Wu[1], Haoming Guo[1]

[1] Institute of Software, Chinese Academy of Sciences, China
{limin, gaoxu, haoming}@nfs.iscas.ac.cn, wubin@iscas.ac.cn
[2] School of Information Technology,Deakin University, Australia
guangyan.huang@deakin.edu.au
[3] College of Engineering and Science, Victoria University, Australia
Jing.He@vu.edu.au
[4] Nanjing University of Finance and Economics, China

Abstract

Since semantic trajectories can discover more semantic meanings of a user's interests without geographic restrictions, research on semantic trajectories has attracted a lot of attentions in recent years. Most existing work discover the similar behavior of moving objects through analysis of their semantic trajectory pattern, that is, sequences of locations. However, this kind of trajectories without considering the duration of staying on a location limits wild applications. For example, Tom and Anne have a common pattern of *Home* → *Restaurant* → *Company* → *Restaurant*, but they are not similar, since Tom works at *Restaurant*, sends snack to someone at *Company* and return to *Restaurant* while Anne has breakfast at *Restaurant*, works at *Company* and has lunch at *Restaurant*. If we consider duration of staying on each location we can easily to differentiate their behaviors. In this paper, we propose a novel approach for discovering common behaviors by considering the duration of staying on each location of trajectories (DoSTra). Our approach can be used to detect the group that has similar lifestyle, habit or behavior patterns and predict the future locations of moving objects. We evaluate the experiment based on synthetic dataset, which demonstrates the high effectiveness and efficiency of the proposed method.

Introduction

With the advance of mobile computing technology and the widespread use of GPS-enabled mobile devices, the location-based services have been improved greatly. Thanks to positioning technologies, a large amount of moving object data are collected and managed in many applications. Some moving objects share the same moving pattern, which reflects the similar lifestyle, habit or behavior in the real world, according to user trajectories. Intuitively, if two users are considered as similar to each other, they should satisfy some common behavior inferred from their trajectories. In this paper, we study data mining techniques for discovering common behaviors of objects.

In recent years, research on trajectory pattern mining has attracted a lot of attentions. It studies the problem of finding the movement behavior or similar lifestyle between moving objects. Trajectory pattern mining can be widely used in many applications. Finding similar users based on trajectory pattern can support friend recommendation to help people to find friends with similar behavior or lifestyle. In the crime analysis domain, finding accomplices through user common behavior search can prevent, decrease and control the criminals. Discovering common behavior can predict the future locations of moving objects, which can enable goods and location recommendation to recommend some goods related to user's next location. Thus, trajectory pattern mining has wide applications.

There exist a lot of researches on trajectory pattern mining in the literatures. In general, the existing researches can be divided into two categories: geographic-based trajectory pattern and semantic-based trajectory pattern. The former one mainly focuses on the geographic features of trajectories, such as shape, direction and speed etc. The latter one discovers trajectory pattern based on the semantic features of trajectories.

In geographic-based methods, trajectories with close distance and similar shape are considered to have grater similarity to group together as a common behavior. In Fig. 1, for example, three persons' trajectories are shown in the geographic layer. From the view of geographic feature, Jim

is more similar to Tom than Anne, because the distance and shape between Tom and Jim are more closer and similar than that of Tom and Anne. However, geographic similarity is lack of semantic information, which cannot discover similar behavior of moving objects.

In semantic-based methods, semantic tags are introduced to capture user's interests and lifestyle. As shown in Fig. 1, the same three persons' semantic trajectories are shown in the semantic layer. However, the current semantic-based methods ignore the duration of staying on each location of trajectories. For instance, in Fig 1, Tom and Anne have a common pattern of *Home* → *Restaurant* → *Company* → *Restaurant*, but they are not similar, since Tom works at restaurant, sends snack to someone at company and return to restaurant while Anne has breakfast at restaurant, works at company and has lunch at restaurant. Thus, the existing semantic-based methods are not suitable to discover common behavior with high accuracy.

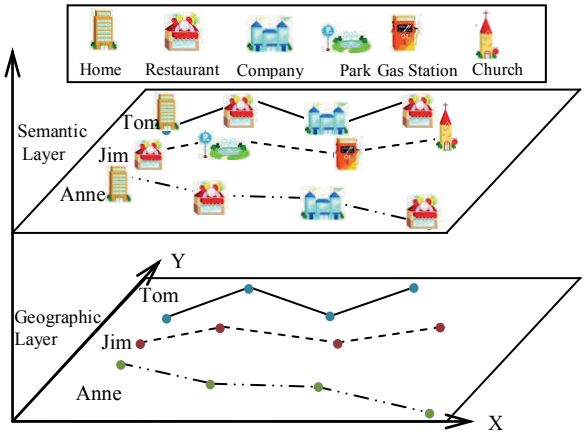


Fig. 1 *Geographic and Semantic Trajectories.*

According to above-mentioned reasons, in this paper, we propose a novel framework, called DoSTra, to discover common behavior of objects considering the duration of staying on each location of trajectories. DoSTra has the advantages that 1) it improves the accuracy to distinguish different behavior patterns, and 2) it can detect the group that has similar lifestyle, habit or behavior patterns. Fig. 2 shows an example of user behaviors with the staying durations. Tom and Anne have the same common pattern as Fig. 1, that is, *Home* → *Restaurant* → *Company* → *Restaurant*, but if we consider the duration of staying on each semantic tag we can easily to differentiate their behaviors. As shown in the figure, since the different duration staying on company and restaurant which reflects their different intentions, Tom and Anne are distinguished to different behaviors.

Given a set of raw trajectories, it's necessary to transform raw trajectories to semantic trajectories, and discover the frequent semantic trajectory patterns of each user with staying durations firstly, each pattern represents one movement behavior of the user. With the duration of staying on each location of trajectories, there is a need to define a new similarity measurement for comparing a pair of patterns between different moving objects. Finally, based on the pattern similarity, we could cluster similar patterns into groups hierarchically, where each cluster represents a group of objects with a set of similar patterns that can infer the common behavior of objects. Based on the above description, the major challenges are summarized as follows:

- Design a new semantic trajectory pattern mining technique with staying durations.
- Define a new pattern similarity measure to enhance the accuracy of the measurement with staying durations.
- Design a new clustering technique to discover common behaviors of objects.

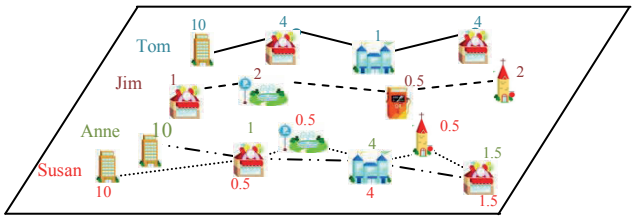


Fig. 2 *User Behaviors with Staying Durations.*

To approach the above challenges, DoSTra provides novel algorithms to discover semantic trajectory patterns, measure pattern similarity and cluster common behavior of objects. In summary, the main contributions of this paper are as followings.

- We define a new semantic trajectory pattern to represent movement behavior with staying durations.
- We propose an effective semantic trajectory pattern mining algorithm.
- We propose a new measurement to evaluate the similarity between patterns with staying durations.
- We propose a novel clustering algorithm based on pattern similarity to discover common behavior.
- We demonstrate the effectiveness and efficiency of our methods on synthetic moving object trajectories.

The remaining of this paper is organized as follows. Related works are discussed in Sect. 2. Our problem is defined in Sect. 3. The three components of DoSTra, including semantic trajectory pattern mining, pattern similarity and similarity-based user clustering, are detailed in Sects. 4, 5 and 6 respectively. The experimental results are shown in Sect. 7. Finally, Sect. 8 concludes the paper.

Related Work

In this section, we first present a survey of methods for trajectory similarity measurement, and then review some existing data mining techniques to find trajectory patterns.

Trajectory Similarity Measurement.

The existing researches on similarity measurement can be divided into two categories: geographic similarity and semantic similarity.

Geographic similarity mainly focuses on the geographic features of trajectories. Most early existing researches on trajectory similarity are Euclidean-based distance measures, DTW (Keogh, 2002), EDR (Chen *et al*., 2005), LCSS (Vlachos *et al*., 2006) are the most representative. In addition, many works measure the similarity between movement behaviors from trajectories (Li *et al*., 2008; Zheng *et al*., 2010). In Hung *et al*. (2011), clue-aware trajectory similarity considers the silent durations. A partition-and-group method (Lee *et al*., 2007) exploits three types of distance measurement. Lu *et al*. (2009) evaluates the greater similarity between trajectories when both the occurrence time and locations are approximate.

Semantic similarity mainly focuses on the semantic features of trajectories. The study of semantic trajectories has been discussed in (Alvares *et al*., 2007; Parent *et al*., 2013; S. Spaccapietra *et al*. 2011; Yan *et al*., 2011). In these literatures, the semantic trajectory is considered as a set of spatio-temporal positions complemented with annotations. According to this way, many applications extract behavior knowledge in semantic aspects than in merely positional data.

In recent years, some work measure the similarity based on semantic trajectories (Ying *et al*., 2010 & 2014; Zheng *et al*., 2011a &2011b & 2009; Liu *et al*., 2012). In these work, each trajectory is transformed into the semantic trajectory first, then evaluate user similarities to recommend potential friends. In Ying *et al*. (2010), the author considers the similarity between two users in terms of the similarity of their maximal semantic trajectory patterns. (Zheng *et al*., 2011a &2011b & 2009) mine a semantic hierarchical tree-structured framework to calculate similarity, In Ying *et al*. (2014), geographic feature and semantic feature are used to predict user's next location.

Since geographic similarity measures only consider the location information, while semantic similarity measures are not designed to handle the staying durations of trajectories, our proposed similarity measurement is different from current methods.

Trajectory Pattern Mining.

The existing work on discovering common pattern could be classified into two categories: trajectory clustering and trajectory pattern mining.

Trajectory pattern mining studies the movement pattern of moving objects, most early researches focus on the spatial-temporal characteristics of trajectories (Tsai *et al*., 2011; Smouse *et al*., 2010; Zhu ,2011; Li *et al*., 2010a). The basic idea of these literatures is to use sequential pattern mining algorithm to discover frequent patterns from transformed trajectories. In Alvares et al. (2007), a semantic trajectory pattern mining method is proposed, which applies a sequential pattern mining algorithm on semantic trajectories.

Because these methods do not consider the duration of staying on each location of trajectories Thus, these methods may not work well when staying durations appear in behavior patters.

Trajectory clustering aims at catching common patterns or behaviors from moving objects. Many works find patterns of travelling together, including snowball (Guo *et al*., 2014), moving cluster (Kalnis *et al*., 2005), flock (Naymat *et al*., 2007), convoy (Jeung *et al*., 2008), compainion (Tang *et al*., 2012) ,swarm (Li *et al*., 2010b) and gathering (Zheng *et al*., 2013). Besides, another kind of clustering focuses on the common paths for a group of moving objects. Partition-and-group framework (Lee *et al*., 2007) partitions each trajectory into a set of sub-trajectories, and then groups clusters using distance function. In Wu *et al*., (2012), both the spatial and temporal criteria are considered for trajectory dividing and clustering.

However, these methods cluster trajectories either from the aspect of individual or group, which may not work well to discover common behavior of objects. Moreover, these approaches are not suitable for semantic trajectory.

Problem Statement

In this section, we give the preliminary concepts that will be used in further discussions in the rest of the paper. Then we show the architecture of our approach. Table.1 lists the notations used throughout this paper.

Table 1: List of Notations.

Notation	Explanation
I	a semantic item
ST	a semantic trajectory
TP	a semantic trajectory pattern
$\mathcal{D}$	a semantic trajectory dataset
A	an annotation
	a duration threshold
	a minimum length threshold
min_sup	a minimum support threshold
$I = I'$	I is item equal to I'
$TP \prec TP'$	TP is semantic contained in TP'
LCS	a longest common sub-sequence
ER	an effective range

Preliminary Concepts

The raw trajectory of a moving object is recorded as a sequence of spatio-temporal points. Thanks to city information, the spatio-temporal points can be transformed to semantic meanings of places of interest, such as shops, restaurants, and cinemas etc. Thus, a semantic trajectory

can be enhanced with annotations from the raw trajectory. Semantic trajectory is the basement of our work, and the rest of the paper is based on it.

Let $I = (A, t)$ be a ***semantic item***, where A is an annotation, t is the staying durations on the annotation of A. Let $ST = \langle I_1, I_2, \ldots, I_n \rangle$ be a ***semantic trajectory*** which is an order sequence of semantic items, where I_i is a semantic item, $1 \leq i \leq n$. As shown in Fig.2, the semantic trajectory of Tom is represented as < (*Home*, 10), (*Restaurant*, 4), (*Company*, 1), (*Restaurant*,4) >.

In order to describe semantic trajectory pattern, we should introduce the definitions of ***item equal***, ***semantic containment*** firstly.

Definition 1 (Item Equal, =): Given two semantic items $I_1 = (A_1, t_1)$ and $I_2 = (A_2, t_2)$, and a duration threshold . I_1 is equal to I_2, denoted by $I_1 = I_2$, if the following conditions are satisfied:

(1) $A_1 = A_2$;

(2) | − |/) ≤ $_t$.

The second condition indicates that the difference ratio between two items' staying durations is limited to .

Definition 2 (Semantic Containment, ≺): Given two sequences of semantic items $S_1 = \langle I_1, I_2, \ldots, I_k \rangle$ and $S_2 = \langle I'_1, I'_2, \ldots, I'_n \rangle$, and a duration threshold , S_1 is semantic contained in S_2, defined by $S_1 \prec S_2$, if and only if there exists a sequence of integers $1 \leq i_1 < \cdots < i_k \leq n$ such that:

$$\forall\, 1 \leq j \leq k, I_j = \quad w.r.t \quad .$$

If two sequences of semantic items S_1 and S_2 satisfy $S_1 \prec S_2$, we also denote that S_1 is a ***sub-sequence*** of S_2.

From the above definitions, the definition of semantic trajectory pattern can be assigned.

Definition 3 (Semantic Trajectory Pattern): Given a set of semantic trajectories $\mathcal{D} = (ST_1, ST_2, \ldots ST_n)$, a sequence of semantic items $TP = \langle I_1, I_2, \ldots, I_k \rangle$, a duration threshold , and a minimum support threshold *min_sup*, *TP* is a semantic trajectory pattern, represented as $I_1 \rightarrow I_1 \rightarrow \cdots \rightarrow I_k$, if and only if:

$$_{\mathcal{D},}\ (\quad) \geq min\_sup$$

where the support $_{\mathcal{D},}$ of a semantic trajectory pattern *TP* is the number of semantic trajectories $ST \in \mathcal{D}$ such that $TP \prec ST$ *w.r.t* .

Without loss of generality, we use letters to represent the annotation of semantic items, and a semantic trajectory dataset are shown in Table 2. Suppose = 0.5 and $s_{sup} = 2$, according to Definition 4, $(a, 10) \rightarrow (b, 0.5) \rightarrow (c, 4) \rightarrow (b, 1.5)$ is a semantic trajectory pattern. However, we can see that any $(a, 10) \rightarrow (b, t) \rightarrow (d, 4) \rightarrow (b, 1.5)$ is a semantic trajectory pattern whenever $t \in [0.5, 1]$. Thus, we use average staying durations to represent the semantic trajectory pattern.

Table 2: An Example of Semantic Trajectory Dataset.

Sid	Semantic Trajectory
1	<(*a*,10),(*b*,4),(*c*,1),(*b*,4)>
2	<(*b*,1),(*d*,2),(*e*,0.5),(*f*,2)>
3	<(*a*,10),(*b*,1),(*c*,4),(*b*,1.5)>
4	<(*a*,10),(*b*,0.5),(*d*,0.5),(*c*,4),(*f*,0.5),(*b*,1.5)>

In order to avoid discover duplicate behaviors, we only maintain the ***maximal semantic trajectory pattern*** to represent user's habit behavior. A semantic trajectory pattern *TP* is a maximal pattern if *TP* cannot be enlarged, which means ∄ ' *s.t. TP*' is a semantic trajectory pattern and $TP \prec TP$'.

According to the definition of the semantic trajectory pattern, the problem in this paper can be formulated as follows.

Problem Statement: Given a set of semantic trajectories and two thresholds, *min_sup* and $_t$, where *min_sup* is for the frequent pattern mining, $_t$ is for time difference ratio. The problem is to discover common behaviors between moving objects with the following steps (1) discover semantic trajectory patterns from semantic trajectories *w.r.t min_sup* and ; and then (2) measure the pattern similarity between moving objects; (3) discover a set of clusters, where each cluster has similar lifestyle, habit or behavior patterns, based on the pattern similarities.

System Overview

In this subsection, we brief an overview of our proposed approach, DoSTra, to discover common behavior. The general framework of DoSTra can be divided into three parts: (1) semantic trajectory pattern mining (*STPM*), (2) pattern similarity (*P-Similarity*) and (3) similarity-based user clustering (*SUC*). Specifically, given a set of raw trajectories, first, we transform raw trajectories to semantic trajectories, and mine semantic trajectory patterns for each moving object based PrefixSpan. Then, we design the pattern similarity based on time-weigth. Finally, we perform a hierarchical clustering method with LCS-boundary pruning strategy, where each cluster represents the common behavior patterns from potential friends.

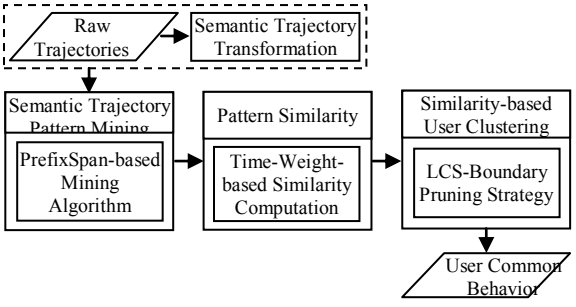


Fig. 3 System Overview.

Semantic Trajectory Pattern Mining

In this section, we provide details of the semantic trajectory pattern mining algorithm. However, we first

have to transform raw trajectories to semantic trajectories, which has been studied in literatures (Alvares *et al.* 2007). The detail of this phase is omitted due to space limitation.

After transforming raw trajectories to semantic trajectories, the major behavior patterns of each moving object can be mined from its semantic trajectories. We propose the STPM algorithm to discover semantic trajectory patterns, which modifies the PrefixSpan method to adapt the staying durations.

The difference between STPM and PrefixSpan is the construction of projected database. For a annotation α, α-projected database consists of subsequences prefixed with the first occurrence of α in PrefixSpan. However, the projection position of STPM is not always the first occurrence of α.

Let $S = < I_1, I_2, \ldots, I_n >$ be a sequence of semantic items, for any $I_i = (A_i, t_i)$, the ***proj-tuple*** of I_i, denoted as , is a triple of ($S.sid$, i, t_i), where $S.sid$ is the sequence id of S, i is the occurrence position of I_i in S, and t_i is the staying durations of I_i. The ***proj-sequence*** of A_i *w.r.t.* in S, denoted as , is the subsequence of S where all items that precede I_i are removed, that is $<I_{i+1}, I_{i+2}, \ldots, I_n>$.

Let $\mathcal{D}$ be a semantic trajectory dataset, α be a annotation, the ***α-projected database*** of $\mathcal{D}$, denoted as $\mathcal{D}$, is a set of couples {(,), (,), …, (,)}, where S_i is the semantic trajectory of $\mathcal{D}$ which contains α, and $_i$ are the corresponding proj-tuple and proj-sequence of α in S_i. Take Table 2 for example, the b-projected database of Table 2 is shown in Table 3, which is ordered in ascending order of durations.

Table 3: b-projected database.

Prefix	Proj-Tuple	Proj-Sequence
b	(4, 2, 0.5)	<(d,0.5),(c,4),(f,0.5),(b,1)>
	(2, 1, 1)	<(d,2),(e,0.5),(f,2)>
	(3, 2, 1)	<(c,4),(b,1)>
	(3, 4, 1.5)	∅
	(4, 6, 1.5)	∅
	(1, 2,4)	<(c,1),(b,4)>
	(1, 4, 4)	∅

Algorithm 1 presents the pseudo code of STPM algorithm. We call STPM(∅, $\mathcal{D}$, *min_sup,*) to discover trajectory patterns of each moving object. In algorithm 1, we first scan α-projected database to find the set of frequent-1 itemset (Line 1-3), then every frequent annotation is appended to α to form α', we construct α'-projected database and sort it by durations in ascending order (Line 4-7), after that we scan proj-tuple of $\mathcal{D}$, from first to end, for every , we calculate the range of durations which covers all items that equal to the one point to, if all the equal items satisfy min_sup, then we generate a new item with average durations, after that the subsets of patterns can be mined by recursively (Line 8-15). If there is no possible to generate any new frequent item, the processing will be terminated, and output I as a qualified pattern (Line 16-17).

Algorithm 1: STPM Algorithm

Input: I: a prefix semantic item; $\mathcal{D}$: the α-projected database, if $\alpha \neq \emptyset$, otherwise, a semantic trajectory dataset $\mathcal{D}$ of an object; *min_sup*: the minimum support threshold; : the time duration threashold.
Output: every qualified maximal semantic trajectory patterns *sq*.

```
1.  find ← 0;
2.  α ← I.A;     // get the annotation of I;
3.  Scan D to find the set of frequent annotations AS w.r.t min_sup;
4.  for each annotation β in AS do
5.      α' ← append β to α;
6.      D' ← construct α'-projected database;
7.      Sort D' in ascending order of durations;
8.      for each couple C ← ( ' , ' , ' ) in D' do
9.          W ← [ ' .duration, T' .duration/ − ]; //range of equal item
10.         sup ← number of couples whose duration is in W;
11.         if sup ≥ min_sup then
12.             I' ← (α', Σ_{=0} [j]. 'duration / );
13.             I_new ← append I' to I;
14.             STPM(I_new, D_{α',W}, min_sup, );
15.             find ← 1;
16. if find = 0 then
17.     output I as a qualified pattern sq;
```

Semantic Pattern Similarity

To identify how similar between two maximal semantic trajectory patterns, we should observe how many common parts the two patterns contain. Thus, we use the concept of the ***Longest Common Sub-sequence*** (LCS) to evaluate the similarity between two patterns, which is similar to (Bergroth *et al.*, 2000), but extends the staying durations.

Let $P_1 = < I_1, I_2, \ldots, I_m >$ and $P_2 = < I'_1, I'_2, \ldots, I'_n >$ be two semantic trajectory patterns, the ***longest common sub-sequence*** of P_1 and P_2 is a longest pattern LCS = < … > which is both a sub-sequence of P_1 and P_2. Specifically, there exist $1 \leq i_1 < \cdots < i_k \leq m$ and $1 \leq j_1 < \cdots < j_k \leq n$ such that: $\forall\, 1 \leq p \leq k$, = and = , where t^*_p = (+ / .

Since LCS contains bias of staying durations, we use a ***time-weight*** to measure the weight of item in LCS.

Definition 4 (Time-Weight): Suppose the description of P_1, P_2 and LCS is as above, the time-weight of in LCS, denoted by ( ), is defined as following:

( ) = 1− | − |/)

In light of the definition of LCS and time-weight, we define the pattern similarity between two patterns as follows.

Definition 5 (Pattern Similarity): Suppose the description of P_1, P_2 and LCS is as above, the similarity between P_1 and P_2, denoted by P-Sim(P_1, P_2), is defined as following:

$$P\text{-}Sim\ (P_1, P_2) = \left(/ | \ | + / | \ | \right) \times \Sigma^{|\ |}$$

For example, suppose = 0.5, given a pattern P = < (a, 10), (b, 1), (c, 4), (b, 1) > and a pattern Q = < (a, 10), (b, 0.5), (d, 0.5), (c, 4), (f, 0.5), (b, 1)>, their longest common sequence is $LCS(P, Q)$ = < (a, 10), (b, 0.75), (c, 4), (b, 1) >. Table 3 gives the time-weight of LCS. Consequently, the pattern similarity between P and Q is $P\text{-}Sim(P,Q)$ = (1/4 +1/6) × (1+0.5+1+1) = 1.46.

Table 3: An Example of Time-Weight.

LCS / *Duration*	I_1 (a, 10)	I_2 (b, 0.75)	I_3 (c, 4)	I_4 (b, 1)
P	10	1	4	1
Q	10	0.5	4	1
(I)	1	0.5	1	1

We modify the longest common sequence algorithm to facilitate the pattern similarity measurement. As we known, dynamic programming is the most popular solution for longest common sequence problem. We use a matrix SM to store the similarity between P and Q at each step of calculation, the calculation formula as follows.

$$SM[i,j]=\begin{cases}0 & \text{if } i=0 \text{ or } j=0\\ \left(\frac{SM[i-1,j-1]}{1/(i-1)+1/(j-1)}+\left(1-\frac{|P[i].t-Q[j].t|}{\max(P[i].t,Q[j].t)}\right)\right)\times\left(\frac{1}{i}+\frac{1}{j}\right) & \text{if } P[i]=Q[j] \; w.r.t.\; \delta_t\\ \max(SM[i-1,j],SM[i,j-1]) & \text{otherwise}\end{cases}$$

The algorithm for pattern similarity is listed in Algorithm 2.

Algorithm 2: P-Similarity Algorithm

Input: P, Q: semantic trajectory patterns; $|P|$, $|Q|$: length of P and Q; : a duration threshold;
Output: SM: $P\text{-}Sim$ of P and Q; $LCSM$: LCS of P and Q;
1. **for** $i \leftarrow 0$ *to* $|P|$ **do**
2. $\quad SM[i, 0] \leftarrow 0$; $LCSM[i, 0] \leftarrow mo\_set \leftarrow$;
3. **for** $j \leftarrow 0$ *to* $|Q|$ **do**
4. $\quad SM[0, j] \leftarrow 0$, $LCSM[0, j] \leftarrow \emptyset$
5. **for** $i \leftarrow 1$ *to* $|P|$ **do**
6. $\quad$ **for** $j \leftarrow 1$ *to* $|Q|$ **do**
7. $\quad\quad$ **if** $P[i] = Q[j]$ *w.r.t* **then**
8. $\quad\quad\quad S \leftarrow SM[i\text{-}1, j\text{-}1]$
9. $\quad\quad\quad rst_1 \leftarrow$ ____ × (/ + /);
10. $\quad\quad\quad rst_2 \leftarrow$ (− ____) × (/ + /);
11. $\quad\quad\quad SM[i, j] \leftarrow rst_1 + rst_2$;
12. $\quad\quad\quad LCSM[i, j] \leftarrow$ append ($P[i].A$, ____) to $LCSM[i, j]$;
13. $\quad\quad$ **else**
14. $\quad\quad\quad S_1 \leftarrow SM[i\text{-}1, j]$; $S_2 \leftarrow SM[i, j\text{-}1]$;
15. $\quad\quad\quad rst_1 \leftarrow$ ____ × (/ + /);
16. $\quad\quad\quad rst_2 \leftarrow$ ____ × (/ + /);
17. $\quad\quad\quad$ **if** $rst_1 < rst_2$ **then**
18. $\quad\quad\quad\quad SM[i, j] \leftarrow rst_2$; $LCSM[i, j] \leftarrow LCSM[i, j\text{-}1]$
19. $\quad\quad\quad$ **else**
20. $\quad\quad\quad\quad SM[i, j] \leftarrow rst_1$; $LCSM[i, j] \leftarrow LCSM[i\text{-}1, j]$
21. **return** $SM[|P|, |Q|]$, $LCSM[|P|, |Q|]$;

In algorithm 2, we use the dynamic programming method to calculate the pattern similarity. First, matrix SM and $LCSM$ are initialized (Line 1-4), which represent similarity and LCS between P and Q respectively. Then we fill SM and $LCSM$ step by step (Line 5-20). Finally, the similarity and LCS between P and Q will be returned (Line 21).

Similarity-based User Clustering

In this section, we describe the similarity-based user clustering (SUC) algorithm for clustering common behavior patterns based on pattern similarity.

Since density-based clustering method may introduce noise, we use a hierarchical clustering method. The basic idea of SUC is a bottom-up algorithm that treats each pattern as a singleton cluster at the beginning and merge pairs of clusters with different moving objects according to similarity until the minimum length threshold satisfied.

Algorithm 3: SUC Algorithm

Input: $MOP = \{mop_1, mop_2, \ldots, mop_N\}$: a set of moving objects patterns, where $mop_i=(i, U, P)$; : a minimum length threshold
Output: *mo_set*: the moving object clusters with common behavior.
1. *mo_set* ← ∅;
2. **for** $i \leftarrow 1$ *to* $N - 1$ **do**
3. $\quad$ **for** $j \leftarrow i + 1$ *to* N **do**
4. $\quad\quad M[i, j].sim$, $M[i, j].lcs \leftarrow$ P-Similarity($mop_i.P$, $mop_j.P$);
5. *stop* ← 0;
6. **while** *stop* = 0 **do**
7. $\quad$ *stop* ← 1;
8. $\quad$ *m_sim* ← 0; *m_i* ← 0; *m_j* ← 0;
9. $\quad$ **for** $i \leftarrow 1$ *to* $N - 1$ **do**
10. $\quad\quad$ **for** $j \leftarrow i + 1$ *to* N **do**
11. $\quad\quad\quad$ **if** $|SM[i, j].lcs| \geq$ && *m_sim* < $SM[i, j]$ && $mop_i.U \cap mop_j.U = \emptyset$ **then**
12. $\quad\quad\quad\quad$ *m_sim* ← $SM[i, j]$;
13. $\quad\quad\quad\quad$ *m_i* ← *i*; *m_j* ← *j*; *stop* ← 0;
14. $\quad$ **if** *stop* = 0 **then**
15. $\quad\quad$ $MOP \leftarrow MOP \backslash mop_{m\_j}$; //remove mop_{m_j} from *MOP*
16. $\quad\quad$ $mop_{m\_i} \leftarrow (m\_i, mop_{m\_i}.U \cup mop_{m\_j}.U, M[m\_i, m\_j].lcs)$;
17. $\quad\quad$ *Remove the m_j-th row from matrix M*;
18. $\quad\quad$ $N \leftarrow N - 1$;
19. $\quad\quad$ **for** $j \leftarrow m\_i + 1$ *to* N **do**
20. $\quad\quad\quad M[m\_i,j].sim$, $M[m_i,j].lcs \leftarrow$P-Similarity($mop_{m\_i}.P$, $mop_j.P$);
21. $\quad\quad$ **for** $j = 1$ *to* $m\_i - 1$ **do**
22. $\quad\quad\quad M[j,m\_i].sim$, $M[j,m_i].lcs \leftarrow$P-Similarity ($mop_j.P$, $mop_{m\_i}.P$);
23. **for** *mop in MOP* **do**
24. $\quad$ **if** $|mop.U| > 1$ **do**
25. $\quad\quad$ *mo_set* ← *mo_set* ∪ *mop*;
26. **return** *mo_set*;

Algorithm 3 shows SUC algorithm. In algorithm 3, we first initialize similarity matrix of initial clusters at the outset (Line 1-5), where each cluster consists of a pattern and a corresponding moving object. Then, we cluster each layer from bottom to up until the minimum length threshold not satisfied (Line 6-22), in each iteration of the clustering process, the most similar pair of patterns between each two moving objects would be merged to form a new cluster (Line 9-13). After that, we check whether there exists a new merged cluster, if so, we would remove the previous clusters, insert the new one and update the similarity matrix (Line 14-22), Finally, clusters are returned (Line 23-26).

From algorithm 3 we can observe that pattern similarity of each pair of patterns between every two moving objects

should be calculated at first, and in each bottom up clustering step, the pattern similarity will be also be calculated N times. However, this is time-consuming and not necessary in some conditions.

Since the semantic trajectory pattern consists of a series of semantic items with duration of staying time. As a result, the similarity of two patterns is constrained by staying time.

Definition 6 (Effective Range): Give a semantic trajectory pattern $P = < I_1, I_2, \ldots, I_k >$, a duration threshold . The effective range of I_i, denoted by Er_i, is defined as following:

$$Er_i = (A_i, [t_{i\_min}, t_{i\_max}])$$

where A_i is the annotation of I_i, $t_{i\_min} = (1-)\times t_i$ and $t_{i\_max} = /(1 -)$. Thus, the effective range of P is $ER_p = <Er_1, Er_2, \ldots, Er_k>$.

Let $Er = (A, [t_{min}, t_{max}])$ be an effective range, $I' = (A', t')$ is an ***effective item*** of Er if $' = A$ and $' \in [t_{min}, t_{max}]$, which is denoted by $I' \propto Er$.

Definition 7 (LCS-Boundary): Give two semantic trajectory patterns P and Q, an effective range $ER_p = <Er_1, Er_2, \ldots, Er_k>$ of P, the LCS-boundary between P and Q, denoted by $LCS\text{-}B(P, Q)$, is the number of items $I' \in Q$ such that $I' \propto Er_i$, where $Er_i \in ER_p$.

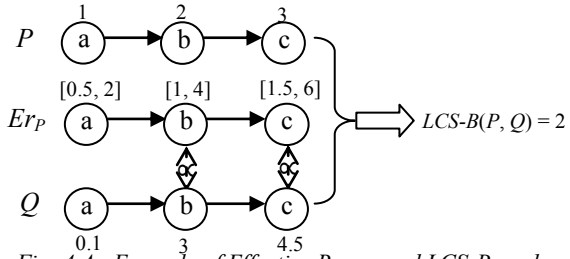


Fig. 4 An Example of Effective Range and LCS-Boundary

As shown in Fig. 4, suppose = 0.5, a semantic pattern $P = <(a, 1), (b, 2), (c, 3)>$, the corresponding effective range of P is $Er_p = <(a, [0.5, 2]), (b, [1, 4]), (c, [1.5, 6])>$, which represents it's impossible to find common sequence of P outside the range of Er_p. Like in pattern Q, $(a, 0.1)$ is not a sub-sequence of P, while $(b, 3)$ and $(c, 4.5)$ are effective items to be sub-sequences of P. Thus, according to the definition 7, the $LCS\text{-}B(P, Q)$ is 2.

Algorithm 4 shows the Optimized-SUC algorithm, which uses LCS-boundary pruning strategy to optimize SUC. The initialization is similar to SUC (Line 1-7). Then, we use pruning strategy to reduce the pattern similarity computation cost when cluster similar patterns (Line 8-31). After that, we check whether there exists a new merged cluster, if so, we would remove the previous clusters, insert the new one and update the similarity matrix (Line 17-31). Since the most time-consuming task is to compute similarity between patterns, then algorithm 4 can prune most of invalid process and improve efficiency.

Algorithm 4: Optimized-SUC Algorithm

Input: $MOP = \{mop_1, mop_2, \ldots, mop_N\}$: a set of moving objects patterns, where $mop_i=(i, U, P)$; : a minimum length threshold

Output: *mo_set*: the moving object clusters with common behavior.

```
1.  mo_set ← ∅;
2.  for i = 1 to N - 1 do
3.      for j ← i + 1 to N do
4.          if LCS-B(mop_i.P, mop_j.P) ≥ then
5.              M[i, j].sim , M[i, j].lcs ← P-Similarity(mop_i.P, mop_j.P);
6.          else
7.              M[i, j].lcs ← 0;
8.  stop ← 0;
9.  while stop = 0 do
10.     stop ← 1;
11.     m_sim ← 0; m_i ← 0; m_j ← 0;
12.     for i ← 1 to N - 1 do
13.         for j ← i + 1 to N do
14.             if SM[i, j].lcs ≥ && m_sim < SM[i, j]
                    && mop_i.U ∩ mop_j.U = ∅ then
15.                 m_sim ← SM[i, j];
16.                 m_i ← i; m_j ← j; stop ← 0;
17.     if stop = 0 then
18.         MOP ← MOP\mop_{m_j}      //remove mop_{m_j} from MOP;
19.         mop_{m_i} ← (m_i, mop_{m_i}.U ∪ mop_{m_j}.U, SM[m_i, m_j].lcs);
20.         Remove the m_j-th row from matrix M;
21.         N = N – 1;
22.         for j ← m_i + 1 to N do
23.             if LCS-B(mop_{m_i}.P, mop_j.P) ≥ then
24.                 M[m_i,j].sim,M[m_i,j].cls=P-Similarity(mop_{m_i}.P,mop_j.P);
25.             else
26.                 M[m_i, j].cls = 0;
27.         for j ← 1 to m_i - 1 do
28.             if LCS-B(mop_{m_i}.P, mop_j.P) ≥ then
29.                 M[j,m_i].sim,M[j,m_i].cls=P-Similarity(mop_j.P,mop_{m_i}.P);
30.             else
31.                 M[j, m_i].cls = 0;
32. for mop in MOP do
33.     if |mop.U| > 1 do
34.         mo_set ← mo_set ∪ mop;
35. return mo_set;
```

Experiment

In this section, we conduct a series of experiments to evaluate the proposed algorithms using synthetic datasets. All the experiments are implemented in C/C++ on Intel(R) Xeon(R) CPU E5-2630 0 @ 2.30GHz machine with 4GB of memory running CentOS release 6.2(Final) using GCC 4.4.6 with optimization -O2.

Experimental Setup

To evaluate the techniques proposed for our DoSTra framework, we use a synthetic semantic trajectory dataset, generated from a set of seed moving objects. Each moving object generates a set of semantic trajectories according to his lifestyle.

In order to investigate the effect of staying durations, two parameters are used to control the behavior: pattern category N_{PC} and the random probability Pr. Specifically, we partition the semantic trajectory patterns into N_{PC} classes, which represent the lifestyle of all objects. Each moving object generates trajectories as the follows: the Pr percentage of trajectories are generated randomly, while the rest are generated according to a part of N_{PC} pattern categories. The main parameters are listed in Table 4.

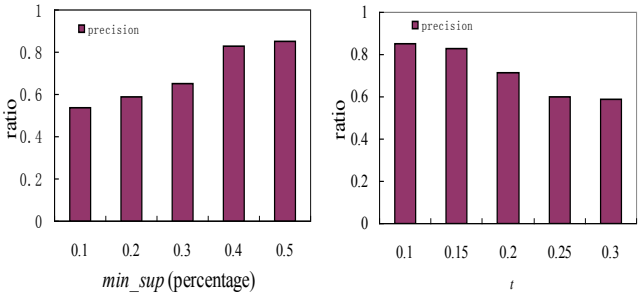
Table. 4. Experiment Settings.

Name	Value	Meaning
N_{mo}	100	the number of moving objects
N_{trj}	1000	the number of semantic trajectories of each mo
$L_{pattern}$	4-12	the average length of each pattern
	01-0.3	the duration threshold
	3	the minimum length threshold
min_sup	0.1-0.5	the minimum support threshold
N_{PC}	20	the number of pattern categories
P_r	0.2	the random probability

Effectiveness Evaluation

We define a metric, called *precision*, to evaluate the effectiveness of our methods: *precision* = $^{+}/^{+}+^{-}$, where $^{+}$ is the number of correct discovered patterns, and $^{-}$ is the number of incorrect discovered patterns.

Fig.5 plots the precision changing with *min_sup* and , the default parameters used in the experiments are: N_{mo} = 100, N_{trj} = 1000, $L_{pattern}$ = 6, N_{PC} = 20 and P_r = 0.2.



(a)Precision Changing with *min_sup* (b) Precision Changing with

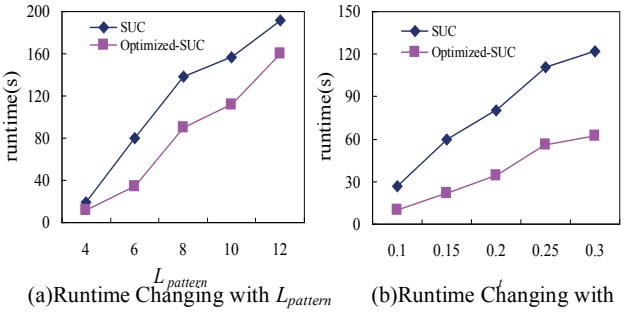
Fig. 5. Precision Evaluation

Fig. 5(a) shows with the increase of *min_sup*, the precision is increase, the reason is that the discovered patterns become more pure with the increase of *min_sup*, then the incorrect discovered patterns is obviously reduced, so the precision is increase. In Fig. 5(b), the precision is decreased with the increase of , since with the increase of , the distinction between different patterns becomes weaker which will lead a rise in error rate.

Efficiency Evaluation

We compare the execution time of SUC and Optimized-SUC by varying the parameters of $L_{pattern}$ and in Fig.6. The default parameters are: N_{mo} = 100, N_{trj} = 1000, *min_sup* = 0.2 (percentage) and = 3.

Fig. 6 shows the performance of SUC and Optimized-SUC changing with $L_{pattern}$ and , where is set to 0.2 in Fig. 6(a) and $L_{pattern}$ is set to 6 in Fig. 6(b). From Fig. 6(a) we can see that with the increase of $L_{pattern}$, runtime of SUC and Optimized-SUC are all increased. The reason is that with the average length of pattern increases, the computation cost of longest common sub-sequence matching also largely increases, which is the main time-consuming part. Similarly, Fig. 6(b) shows that the runtime of two algorithms are increased with the increase of . Because the complexity of longest common sub-sequence matching is increased with the increase of . Moreover, compared with SUC, the performance of Optimized-SUC is greatly improved in both two figures, which shows the validity of our pruning strategy.



(a)Runtime Changing with $L_{pattern}$ (b)Runtime Changing with

Fig. 6. Runtime Evaluation

In summary, all the experiments demonstrate the performance of our algorithms in terms of effectiveness and efficiency. DoSTra can provide acceptable execution performances with high recognition accuracy from different behaviors.

Conclusion

In this paper, we propose the DoSTra framework to discover common behaviors between moving objects. In order to improve the accuracy to distinguish different behavior patterns, we introduce the staying durations. DoSTra mainly consists of three components: semantic trajectory pattern mining, semantic pattern similarity and similarity-based user clustering. Experimental results show that DoSTra is able to effectively and efficiently discover common behaviors from semantic trajectories. For future work, we plan to design the parallelization technique to support massive data.

Acknowledgments

This research is supported by the National Natural Science Foundation of China under Grant No. 61402449, 91124001, 71372188, 61300213, and the Priority Academic Program Development of Jiangsu Higher Education Institutions (PAPD, Food Safety and Engineering).

References

C.-C. Hung, W.-C. Peng, and W.-C. Lee. Clustering and aggregating clues of trajectories for mining trajectory patterns and routes. VLDB Journal, 2011.

C. Parent, S. Spaccapietra, C. Renso, et al. Semantic Trajectories Modeling and Analysis. ACM Computing Surveys, 45(4), 2013.

E.H.-C. Lu, and V.S. Tseng. Mining Cluster-based Mobile Sequential Patterns in Location-Based Service Environments. In MDM, 273-278, 2009.

E.J. Keogh. Exact indexing of dynamic time warping. In VLDB, 406-417, 2002.

G. Al-Naymat, S. Chawla, and J. Gudmundsson. Dimensionality reduction for long duration and complex spatio-temporal queries. In SAC, 2007.

H. Jeung, H. T. Shen, and X. Zhou. Convoy queries in spatio-temporal databases. In ICDE, 2008.

H. Liu, and M. Schneider. Similarity Measurement of Moving Object Trajectories. In IWGS, 2012.

H. P. Tsai, D. N. Yang, M. S Chen. Mining Group Movement Patterns for Tracking Moving Objects Efficiently. IEEE Transactions on Knowledge and Data Engineering, 23(2): 266-281, 2011.

H.-R. Wu, M.-Y. Yeh, M.-S. Chen. Profiling Moving Objects by Dividing and Clustering Trajectories Spatiotemporally. In TKDE, 2012.

J.-C. Ying, H.-S. Chen, K. W. Lin, et al. Semantic trajectory-based high utility item recommendation system. Expert Systems with Applications. 4762-4776, 2014.

J.-G. Lee, J. Han and K.-Y. Whang. Trajectory Clustering: A Partition-and-Group Framework. In SIGMOD, 593-604, 2007.

J. J.-C. Ying, E. H.-C. Lu, W.-C. Lee, et al. Mining User Similarity from Semantic Trajectories. In LBSN, 19-26, 2010.

K. Zheng, Y. Zheng, and N. J. Yuan. On Discovery of Gathering Patterns from Trajectories. In ICDE, 2013.

L. A. Tang, Y. Zheng. On discovery of traveling companions from streaming trajectories. In ICDE, 2012.

L. Bergroth, H. Hakonen, T. Raita. A Survey of Longest Common Subsequence Algorithms. In SPIRE, 39-48, 2000.

L. Chen, M. Ozsu, and V. Oria. Robust and fast similarity search for moving object trajectories. In SIGMOD, 491–502, 2005.

L. Guo, G. Huang, Z. Ding. Efficient Detection of Emergency Event from Moving Object Data Streams. In DASFAA, 422-437, 2014.

L. O. Alvares, V. Bogorny, B. Kuijpers, et al. Towards semantic trajectory knowledge discovery. Data Mining and Knowledge Discovery, 2007.

F. X Zhu. Mining Ship Spatial Trajectory Patterns from AIS Database for Maritime Surveillance. In ICEMMS, 772-775, 2011.

M. Vlachos, M. Hadjieleftheriou, D. Gunopulos, and E.J. Keogh. Indexing multidimensional time-series. VLDB Journal, 15(1), 1-20, 2006.

P. E. Smouse, S. Focardi, P. R. Moorcroft, et al. Stochastic Modeling of Animal Movement. Philosophical Transactions of the Royal Society of London - Series B: Biological Sciences, 365(1550): 2201–2211, 2010.

P. Kalnis, N. Mamoulis, and S. Bakiras. On discovering moving clustering moving clusters in spatio-temporal data. In SSTD, 2005.

Q. Li, Y. Zheng, X. Xie, et al. Mining User Similarity Based on Location History. In Sigspatial GIS, 2008.

S. Spaccapietra, and C. Parent. Adding meaning to your steps. In ER. 13-31, 2011.

Y. Zheng, L. Zhang, and X. Xie. Recommending friends and locations based on individual location history. ACM Transaction on the Web, 2010.

Y. Zheng and X. Xie. Learning Ttravel Recommendations from User-generated GPS Traces. ACM Transactions on Intelligent Systems and Technology (TIST), 2(1), 2011a.

Y. Zheng, L. Zhang, Z. Ma, et al. Recommending Friends and Locations Based on Individual Location History. TWEB, 5(1), 2011b.

Y. Zheng, L. Zhang, X. Xie, et al. Mining Interesting Locations and Travel Sequences from GPS Trajectories. In WWW, pp791-800, 2009.

Z. H. Li, M. Ji, J. G. Lee, et al. MoveMine: Mining Moving Object Databases. In SIGMOD, 1203-1206, 2010a.

Z. Li, B. Ding, J. Han, et al. Swarm: mining relaxed temporal moving object clusters. In VLDB, 2010b.

Z. Yan, D. Chakraborty, C. Parent, et al. SeMiTri: A Framework for Semantic Annotation of Heterogeneous Trajectories. In EDBT, 259-270, 2011.