Discovery of Chasing Patterns in Trajectory Data

Huaqing Wu[1], Weizhuo He[1,2], Qizhi Liu[1](✉), and Yu Yang[3]

[1] State Key Laboratory for Novel Software Technology, Nanjing University, Nanjing, China
lqz@nju.edu.cn
[2] HUAWEI Corporation, Nanjing, China
[3] Pancar Technology Ltd., Nanjing, China

Abstract. Trajectory data contains abundant temporal and spatial information of moving objects. By calculating and analyzing trajectory data, behavior patterns of mobile objects can be found. Among these behavior patterns the discovery of chasing patterns is interesting which will provide effective services for intelligent urban traffic management, criminal investigation, environmental monitoring and other application fields. However, existing chasing pattern discovery methods only compare the local similarity of trajectories, and do not consider the potential significance of stay points or change in speed. This article proposes a new algorithm for chasing pattern discovery based on the stay point detection technology. An optimized stay point detection algorithm is also designed to improve the distance calculation method. The algorithms were evaluated with real and simulation trajectory data and achieved very good results which have higher accuracy and recall rate compared with the existing chasing pattern discovery algorithms.

Keywords: Trajectory data · Chasing patterns · Stay point detection · Great circle distance

1 Introduction

The widespread use of modern positioning technology has produced a large amount of trajectory data. The trajectory data contains rich spatio-temporal information. An in-depth analysis of the spatio-temporal information in the trajectory data can discover the behavior patterns of moving objects, and can produce interesting and beneficial application effects in multiple fields. For example, Alvares et al. proposed an algorithm to detect avoidance behavior pattern that identifies when a moving object is avoiding specific spatial regions, such as security cameras [1]. Jeung et al.'s research on animal group movement pattern discovery can be used for migration event recognition [2]. Siqueira and Bogorny presented formal definitions and an algorithm to find chasing behavior in moving object trajectories [3]. Kang et al. used the chasing pattern to explain the behavioral characteristics of a class of players in the football game, and formulated a new strategy for subsequent games by observing the chasing pattern of the athletes [4]. Among various behavior patterns, we believe that chasing pattern is a very important

Y. Nah et al. (Eds.): DASFAA 2020 Workshops, LNCS 12115, pp. 47–59, 2020.
https://doi.org/10.1007/978-3-030-59413-8_4

pattern, and its discovery technology can be used in applications such as urban traffic management, driverless, security, etc., and can prevent some pre-planned abduction or terrorism to reduce the occurrence of potentially malignant events.

The chasing pattern involves many spatio-temporal information such as the position, time stamp, speed, direction and change trend of the pair of moving objects in the trajectory data, and its discovery process is relatively complicated. This article specializes in chasing pattern discovery methods, and it is the first method that based on stay point detection technology. Because chasing and chased target often share similar staying characteristics, converting the original GPS trajectory of the moving object into a trajectory with stay points will not only help the discovery of the chasing pattern, improving the recall rate, but also help to find more complete chasing pattern. Meanwhile, it can also be used to compress the trajectory data, reducing the calculation amount of pattern discovery. The main contributions of this article are as follows:

- We establish a chasing pattern discovery model based on the detection of stay points. The model can effectively find chasing patterns in trajectories with large angle changes. The detection of stay points in the model helps to screen out trajectories that are more likely to have chasing patterns. The candidate trajectory set can also improve the efficiency of subsequent calculations.
- We design a chasing pattern discovery algorithm. And the stay point detection algorithm is optimized to improve the accuracy of similar trajectory matching and chasing pattern discovery.
- We verified the effectiveness of the above model and the performance of the algorithms through a series of experiments.

In the rest of this article, we give the definition of the problem and the chasing pattern discovery model in the second section. In the third section, we describe the optimization algorithm for stay point detection and the chasing pattern discovery algorithm based on stay point detection. The fourth section introduces the experimental settings and the results of each experiment. The fifth section introduces related work. Finally, we summarizes the full text and discusses future work directions.

2 Chasing Pattern Discovery Model

A trajectory is the path of a moving object and can be described as a function of time. It corresponds to a set of trajectory point data sequences including time stamps and spatial information [5], where the spatial information generally includes the latitude and longitude of the trajectory points, and sometimes includes the altitude of the point. The sequence often contains other information such as the moving speed when the moving object is passing the trajectory point. In this section we give the definition of the original trajectory used in this article, which helps us design a chasing pattern discovery model.

Definition 1 (Original trajectory). *An original trajectory T is a sequence of trajectory points, that is, $T: p_1 \rightarrow p_2 \rightarrow \ldots \rightarrow p_n$.* Each trajectory is uniquely identified by the number *tid*. *The trajectory point $p_i (1 \leq i \leq n)$ is a multivariate group including at least four kinds of information: latitude, longitude, velocity, and timestamp, that is,*

p_i : $\{lat_i, lng_i, speed_i, t_i\}$. The time interval between two consecutive trajectory points does not exceed the time threshold Δt, that is, $0 < t_{i+1} - t_i < \Delta t$.

2.1 Baseline Model

Given a set of candidate original trajectories $\mathbb{T}\mathbb{c}$ and an original trajectory T ($T \notin \mathbb{T}\mathbb{c}$), the problem of finding the chasing pattern is to match T' (($T' \in \mathbb{T}\mathbb{c}$)) with Tone by one. The solution model is:

Given two trajectories T and $T^{'}$, $T : p_1 \rightarrow p_2 \rightarrow \ldots \rightarrow p_n, T^{'} : q_1 \rightarrow q_2 \rightarrow \ldots \rightarrow q_m$, with the average velocity v_1, v_2 respectively. When the following conditions are met, we have a chasing where T is chasing $T^{'}$. That is, there is a chasing pattern between T and $T^{'}$:

(1) For the start and end positions in the two trajectories, there are $distance(p_1, q_1) \leq \varepsilon$ and $distance(p_n, q_m) \leq \varepsilon$, where ε is a small value;
(2) For the average distance threshold Δd, there exists $\sum_{i=1}^{n} distance(p_i, T_2)/n \leq \Delta d$ and $\sum_{j=1}^{m} distance\big(q_j, T_1\big)/m \leq \Delta d$;
(3) For the time difference tolerance Δtd, there exist $0 \leq t_{q_1} - t_{p_1} \leq \Delta td$ and $0 \leq t_{q_m} - t_{p_n} \leq \Delta td$;
(4) For the duration parameter τ, there exists $t_{q_m} - t_{p_1} \geq \tau$;
(5) For the average velocity threshold $\alpha \in [0, 1)$, there exists $(1 + \alpha) \geq (v_1/v_2) \geq (1 - \alpha)$.

Among them, constraint condition (5) is optional. For chasing patterns considering speed, the similarity of the average velocity needs to be calculated.

2.2 Optimization Model

Moving objects usually stop moving when they encounter obstacles, events or places of interest during the journey. Those stay information will be implicit in the trajectory data. For example, a significant reduction in speed can cause several spatial attributes of consecutive trajectory points to be similar, but they still differ in time attributes. By analyzing the trajectory data, several trajectory points corresponding to the stay in the original trajectory can be replaced by one stay point (as shown in Fig. 1).

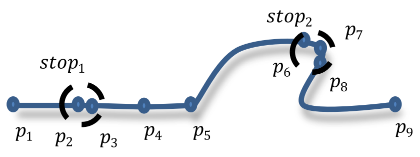

Fig. 1. Stay points

Definition 2 (Stay point). *A stay point* $stop_k : p_{k_1} \rightarrow p_{k_2} \rightarrow \ldots \rightarrow p_{k_n} (1 \leq k_1 < k_n \leq n)$ *corresponds to a segment of the original trajectory* T*, where the distance between* p_{k_1} *and* p_{k_n} *is less than the threshold* Δds*, and* $t_{k_n} - t_{k_1}$ *is greater than the time threshold* Δts*, corresponding to a geographical area where the moving object stays for a period of time.*

We assume that similar stay behaviors are often found in trajectory pairs with chasing patterns and with the stay point detection technology [6] we can find chasing patterns more effectively in trajectory data. Based on this idea, we design a chasing pattern discovery optimization model, as shown in Fig. 2.

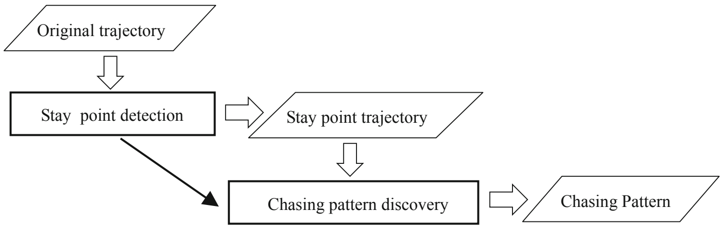

Fig. 2. Frame of the chasing pattern discovery optimization model

In the model, the trajectory segments of all the stay regions in the original trajectory are converted into stay points. Thus forms a new trajectory sequence with stay points, which looks like $T^s : p_1 \rightarrow stop_1 \rightarrow p_4 \rightarrow p_5 \rightarrow stop_2 \rightarrow p_9 \rightarrow \ldots \rightarrow p_n$. When searching for a chasing pattern, we first consider the stay point as an ordinary trajectory point to judge whether there is chasing pattern according to the basic model; then we take the stay points as special trajectory points and use the point-to-point comparison method to identify the true chasing pattern based on the number and distance of the stay points.

3 Chasing Pattern Discovery Algorithm

In the chasing pattern, the track of the chaser is roughly the same as that of the chased, so the chaser may encounter the same events and has similar staying behavior as the chase. Therefore, the stay point is an important feature that can reflect the chasing behavior pattern. Moreover, the trajectory data scale with stay points obtained after using the stay point detection technology is smaller than the original trajectory, which can reduce the amount of calculation for chasing pattern discovery.

3.1 Stay Point Detection Algorithm

Existing stay point detection methods are usually based on the velocity change information in the trajectory [7], or whether the distance between points is less than the distance threshold [8]. This velocity change or distance threshold is only valid in a specific application scenario and a specific time range. In this article, a new stay point detection algorithm (nSPD) is designed to find the chasing patterns considering the time threshold, distance threshold, and the switching times of detection. The algorithm idea is shown in Fig. 3.

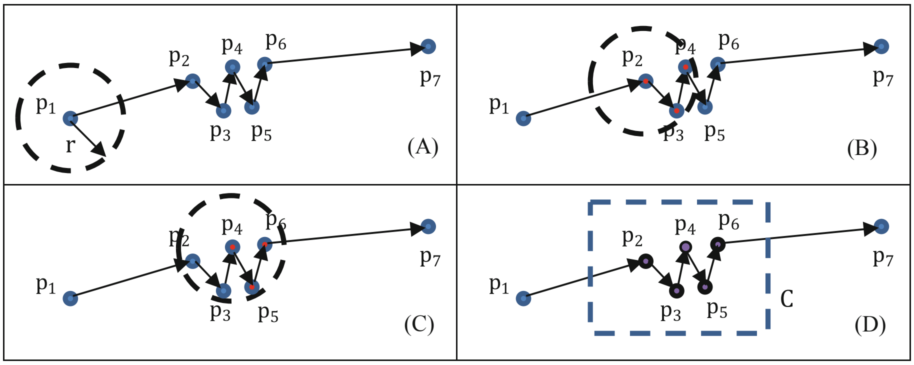

Fig. 3. Stay point detection schematic diagram

Given a trajectory $T : p_1 \rightarrow p_2 \rightarrow \ldots \rightarrow p_7$, we first calculate the physical distance between each point and the point behind it in turn, until the distance between this point and a subsequent point exceeds a given distance threshold r. In Fig. 3(A), $\mathrm{dist}(p_1, p_2) > r$ and no other points that constitute a stay point with p_1 are found, so the center point is moved to p_2, as shown in Fig. 3(B). And $\mathrm{dist}(p_2, p_3) < r$, $\mathrm{dist}(p_2, p_4) < r$, $\mathrm{dist}(p_2, p_5) > r$, i.e. $\mathrm{dist}(p_2, p_3)$ meets the distance threshold as well as $\mathrm{dist}(p_2, p_4)$does, so if the time interval between p_2 and p_4 is not less than the time threshold Δt_s, i.e. $\mathrm{interval}(t_2, t_4) \geq \Delta t_s$, we take p_2, p_3, p_4 as a small cluster and move the center to p_4, as shown in Fig. 3(C). Then $\mathrm{dist}(p_4, p_5) < r$, $\mathrm{dist}(p_4, p_6) < r$, so p_2, p_3, p_4, p_5, p_6 compose a cluster, i.e. the stay point C, as shown in Fig. 3(D). The coordinates of the stay point are obtained by calculating the geometric mean of the coordinates of those points. After that, the original trajectory sequence $\{p_1 \rightarrow p_2 \rightarrow p_3 \rightarrow p_4 \rightarrow p_5 \rightarrow p_6 \rightarrow p_7\}$ becomes the sequence $\{p_1 \rightarrow stop_c \rightarrow p_7\}$. The original 7 points are reduced to 3 points.

In general, for a series of consecutive points $P : \{p_{k_1} \rightarrow p_{k_2} \rightarrow \ldots \rightarrow p_{k_n}\}(1 \leq k_1 < k_n \leq n)$ in an original GPS trajectory $T : p_1 \rightarrow p_2 \rightarrow \ldots \rightarrow p_n$, the nSPD algorithm determines whether P can be regarded as a stay point $stop_k$ according to three parameters, i.e. the time threshold Δt_s, the distance threshold r and the number of times of detection threshold θ_{times}. When $\mathrm{dist}(p_{k_n}, p_{k_1}) \leq r$ and $|t_{k_n} - t_{k_1}| \geq \Delta t_s$, it means that P can be represented by the stay point $stop_k$, the corresponding virtual position of P. And θ_{times} is used to decide whether to add subsequent points p (satisfying $\mathrm{dist}(p, p_{k_n}) \leq r$) to P. Reducing the value of θ_{times} can improve the accuracy of stay point detection.

The stay point is expressed by a quaternion $stop_k : \{lat_k, lng_k, arvt_k, levt_k\}$, where:

$$lat_k = \sum_{k_i=k_1}^{k_n} \frac{lat_{k_i}}{|P|}, k_1 \leq k_i \leq k_n \quad (1)$$

$$lng_k = \sum_{k_i=k_1}^{k_n} \frac{lng_{k_i}}{|P|}, k_1 \le k_i \le k_n \quad (2)$$

In the formula, lat_k represents the average latitude of set P and lng_k represents the average longitude of set P. The time of entering the stay point $arvt_k$ equals t_{k_1}. The time of leaving the stay point $levt_k$ equals t_{k_n}. And average velocity can be add to $stop_k$: $\{lat_k, lng_k, arvt_k, levt_k, v_k\}$ if needed.

Algorithm 1: Stay Point Detection Algorithm (nSPD)

```
Input: original trajectory $\boldsymbol{T}$, threshold $\Delta \boldsymbol{t_s}$, $\boldsymbol{r}$ and $\boldsymbol{\theta_{times}}$
Output: the corresponding stay point trajectory $\boldsymbol{T^s}$
1   $\boldsymbol{p} = \boldsymbol{T}.\mathbf{firstPoint}()$;
2   while $\boldsymbol{p}$ != null
3       $\boldsymbol{p'} = \boldsymbol{p}.\mathbf{nextPoint}()$;
4       $\boldsymbol{times} = \mathbf{0}$;
5       while $\boldsymbol{p'}$ != null
6           if $||\boldsymbol{p} - \boldsymbol{p'}||_{gc} \le \boldsymbol{r}$
7               $\boldsymbol{p}.\mathbf{addStPt}(\boldsymbol{p'})$;
8               $\boldsymbol{p'} = \boldsymbol{p'}.\mathbf{nextPoint}()$;
9           else if $\boldsymbol{times} < \boldsymbol{\theta_{times}}$
10              $\boldsymbol{p} = \boldsymbol{p}.\mathbf{lastStP}()$;
11              $\boldsymbol{times}$++;
12          else if $|\boldsymbol{p}.\mathbf{stPts}()| > \mathbf{0}$ and $\boldsymbol{p}.\mathbf{lastStP}().\boldsymbol{t} - \boldsymbol{p}.\boldsymbol{t} > \Delta \boldsymbol{t_s}$
13              $\boldsymbol{T^s}.\mathbf{add}((\boldsymbol{p}.\mathbf{stPts}()))$;
14              break;
15          else
16              $\boldsymbol{T^s}.\mathbf{add}(\boldsymbol{p})$;
17              $\boldsymbol{p} = \boldsymbol{p'}$;
18              break;
19  return $\boldsymbol{T^s}$;
```

In the algorithm, p.stPts() is the stay point when p is the pivot point, and p.lastStP() is the last point in point p. When calculating the distance between points, we use the great circle distance [9]. The great circle distance from point p to p' is $||p - p'||_{gc}$. Considering that the region of stay points is small, we use the *haversine* formula to calculate the great circle distance.

3.2 Chasing Pattern Discovery Algorithm Based on Stay Point Detection

Given the candidate original trajectory set $\mathbb{T}_{\mathbb{C}}$, if a given original trajectory T ($T \notin \mathbb{T}_{\mathbb{C}}$) with the starting point p_1 is chased by an original trajectory T' T' ($T' \in \mathbb{T}_{\mathbb{C}}$), then T' should pass p_1. In fact, any point in T can be considered as a starting point. So we first align all the trajectories in $\mathbb{T}_{\mathbb{C}}$ with T, intercepting the remaining part of the trajectories after p_1 (including point p_1), then convert the candidate original trajectories in $\mathbb{T}_{\mathbb{C}}$ into the stay point trajectories one by one, thus forming $\mathbb{T}_{\mathbb{C}}{}^s$, and convert T into T^s.

After that, we find which stay point trajectory T_i in $\mathbb{T}\mathbb{c}^s$is chasing T^s, according to the time difference tolerance Δtd, the average velocity threshold α, the duration parameter τ, as well as the number of the stay points and the distance between the stay points int T_i and T^s, thus forming $\mathbb{R}^s$. The specific steps of chasing pattern discovery algorithm (sChPD) based on stay point detection are shown in Algorithm 2.

In the algorithm, the constraint on the velocity (12 lines) is optional. $P_s.t_{end}$ is the end time of the chasing trajectory pair, and $P_s.t_{start}$ is the start time of the chasing trajectory pair.

Algorithm 2: Stay-point-based Chasing Pattern Discovery Algorithm (sChPD)

```
Input: candidate stay points trajectory set $\mathbb{T}\mathbb{c}^s$, stay points trajectory $\boldsymbol{T^s}$, threshold $\boldsymbol{\Delta td}$, $\boldsymbol{\alpha}$ and $\boldsymbol{\tau}$
Output: the corresponding chasing stay points trajectory set $\mathbb{R}^s$
1   Initialize $\mathbb{R}^s$;
2   for $\boldsymbol{T_i}$ in $\mathbb{T}\mathbb{c}^s$
3       $\boldsymbol{p_1 = T^s.\mathrm{firstPoint}()}$;
4       $\boldsymbol{p_2 = T_i.\mathrm{firstPoint}}()$;
5       while $\boldsymbol{p_1} != \boldsymbol{null}$ and $\boldsymbol{p_2} != \boldsymbol{null}$
6           if $\boldsymbol{p_1.arvt} < \boldsymbol{p_2.arvt}$
7               if $||\boldsymbol{p_1} - \boldsymbol{p_2}||_{gc} \le \boldsymbol{\varepsilon}$ and $|\boldsymbol{p_1.arvt} - \boldsymbol{p_2.arvt}| < \Delta \boldsymbol{td}$
8                   $\boldsymbol{P_s.\mathrm{add}(p_2)}$;
9           $\boldsymbol{p_1} = \boldsymbol{p_1.\mathrm{nextStayPoint}}()$, $\boldsymbol{p_2} = \boldsymbol{p_2.\mathrm{nextStayPoint}}()$;
10      if $(\boldsymbol{1+\alpha}) \ge |\boldsymbol{T^s.v/T_i.v}| \ge (\boldsymbol{1-\alpha})$
11          if $\boldsymbol{P_s.t_{end}} - \boldsymbol{P_s.t_{start}} > \boldsymbol{\tau}$
12              $\mathbb{R}^s.\boldsymbol{\mathrm{add}(T_i.t_{id}, P_s)}$;
13      $i$++;
14  return $\mathbb{R}^s$;
```

4 Experimental Evaluation and Algorithm Analysis

In this section, the performance of the proposed stay points detection method and the algorithm of discovering chasing pattern is evaluated with real taxi dataset.

4.1 Settings

Dataset. We use the S-Taxi, trajectories of 109 taxis in a city in northeast China in one day. The size of the dataset is about 168 MB, including 780,000 GPS track sampling points, and its sampling time is 3–10 s after cutting and preprocessing. The points of each trajectory consist of the following information: timestamp, latitude, longitude, and speed of the vehicle when it passed the current position. In addition, we use time intervals to cut and divide the trajectory to facilitate pattern mining and discovery.

Implementations. All experiments are performed on a computer with Intel Core i5-4200 CPU with 8 GB RAM running windows 10, and all the methods are implemented in Python 3.6.

Benchmark Algorithm. Chasing pattern discovery algorithm based on original points.

4.2 Evaluation of Stay Points Detection Algorithm

In the stay points detection algorithm, the parameters that affect the number of stay points detections are mainly the value of the switch time θ_{times}, the time threshold Δt_s and the distance threshold r. In Fig. 4, stay points (SP) percent represents the number of trajectories with stay points as a percentage of the total trajectories.

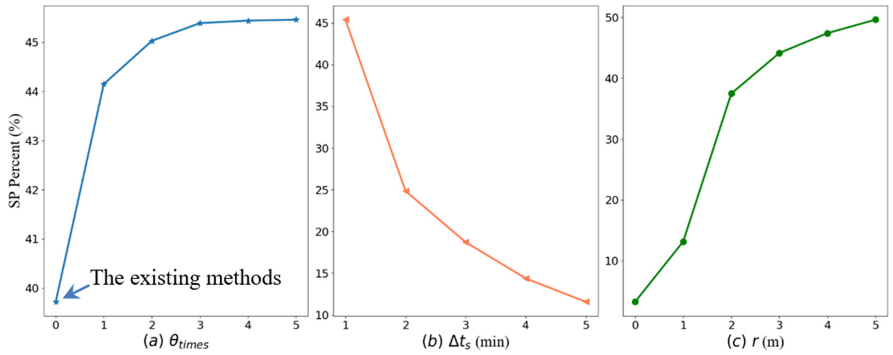

Fig. 4. Stay points (SP) percent on different parameters

The existing stop point detection methods do not use the switch time parameter, which can be considered as the case where the switch time θ_{times} is set to 0 in our proposed method. Regarding the switch time, it can be seen from Fig. 4(a) that there is a large difference between the case where the value of switch time is zero and non-zero, which means that the introduction of the switch time helps the discovery of stay points, and the increase in the value of switch time can be used to find more stay points in the trajectory. But limited by the dataset and other constraints, when the value of switch time is greater than 3, basically no more trajectories with stay points will be found. As for the time threshold, it can be seen from Fig. 4(b) that with the increase of the time threshold, the number of trajectories with stay points is found to decrease, which indicates that the increase of the time threshold effectively filters the set of stay points with small time intervals. However, if the threshold is too large, the normal stay points can also be filtered out. As for the distance threshold, it can be seen from Fig. 4(c) that the increase of the distance threshold can relax the conditions of stay points detection, so that more stay points can be detected. But as the distance threshold increases, those points that are not actually stay points are gathered in a set of stay points, which can cause a certain error.

4.3 Evaluation of Chasing Pattern Discovery Algorithm

Running Time. Figure 5 shows the running time of the two chasing pattern discovery algorithm in different time intervals. These two algorithms are based on the stay points and the original points. The time interval in the figure represents the intervals separated by the time of a day according to the given interval threshold, which is used to divide the trajectory. For example, the interval length in Fig. 5 is set to 30 min, then the trajectory

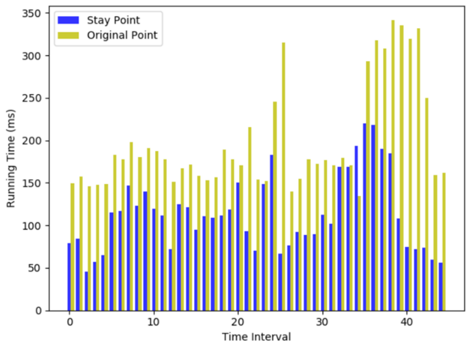

Fig. 5. Running time on different time interval

data of a day will be divided into sub-trajectory according to time and fall into 48 intervals (24 h have 48 time intervals with 30 min length).

In Fig. 5, we set distance threshold $\Delta d = 3$ m, time threshold $\Delta td = 2$ min, average rate threshold $\alpha = 0.5$, and duration $\tau = 5$ min. It can be seen that the chasing pattern discovery algorithm based on the stay points is more efficient in time running than the chasing pattern discovery algorithm based on the original point. This shows that the setting of the stay points shortens the time of finding trajectory with the chasing pattern while reducing the number of trajectory points.

The Number of Chasing Pattern Trajectories(CPT). In Table 1, we selected the trajectory data for 10 time intervals as the candidate trajectory set. The interval length of the divided trajectory is set to 30 min, and other parameter settings are the same as Fig. 5. Then we search the chasing pattern trajectories corresponding to each trajectory from the candidate set. Here we define the statistical method of the number of chasing pattern trajectories as follows. For each trajectory T_i in the given time interval, as long as there is another trajectory T_j that satisfies the condition of chasing T_i, then the total number of chasing pattern trajectories in the given time interval increases by 1. It can be seen from Table 1 that there are more chasing pattern trajectories found by the stay-points-based algorithm (CPT-SP) than that found by the original-points-based algorithm (CPT-OP) in most time intervals.

In order to find the effect of the interval length of the divided trajectory on the chasing pattern discovery, we set the interval length of the divided trajectory to 15 min, 30 min, and 60 min for experimental verification. The results are shown in Table 2. We know that as the interval length increases, the trajectory in each interval will contain more points, but the total number of trajectories will decrease. So the percent of CPT found is increasing along with the interval length increasing. From Table 2, we find that at each kind of interval length, the algorithm based on stay points always finds more trajectories of the chasing pattern than the algorithm based on original points in general.

Parameter Evaluation. Through experiments we find that the main parameters that affect the chasing pattern discovery algorithm are the distance threshold Δd and the chasing duration τ. We chose a 30-min time interval as the experimental data, which contains 324 trajectories. Except for the control parameters, other parameters are set the

Table 1. The number of CPT on different intervals

Interval	CPT-SP	CPT-OP	Total trajectory
0:00–0:30	**237**	231	324
1:00–1:30	**233**	225	312
2:00–2:30	**235**	231	308
3:00–3:30	**222**	218	280
4:00–4:30	**225**	214	296
5:00–5:30	**227**	221	304
6:00–6:30	**275**	265	356
7:00–7:30	**274**	270	364
8:00–8:30	279	279	376
9:00–8:30	285	287	380

Table 2. The percent of CPT on different interval length

Interval Length	CPT-SP Percent	CPT-OP Percent	Total Trajectory
15	0.72959	0.72325	32972
30	0.75863	0.74833	16796
60	0.78155	0.77608	7132

same as Fig. 5. As for the distance threshold parameter, it can be seen from Fig. 6(a) that increasing the distance threshold will find more chasing pattern trajectories. However, this does not mean that the larger the distance threshold is, the better the algorithm will perform, because for two trajectories that are far away and there is no chasing pattern between them, they will be misjudged as having a chasing pattern, which will cause errors. As the distance threshold increases, the differences between the original-points-based method and the stay-points-based methods will reduce. In extreme cases the original-points-based method even find more chasing pattern trajectories.

As for the duration, it can be seen from Fig. 6(b) that the increase in the duration of the chasing pattern trajectory will reduce the number of chasing pattern trajectories found. This shows that the duration of the trajectory with the chasing pattern will not be very long due to the limitation of other parameters such as the distance threshold, and increasing the duration will filter out some trajectories that have been found. In addition, when the duration τ is relatively short, the method based on the original points will have more semantic information than the method based on the stay points, so more chasing pattern trajectories will be found.

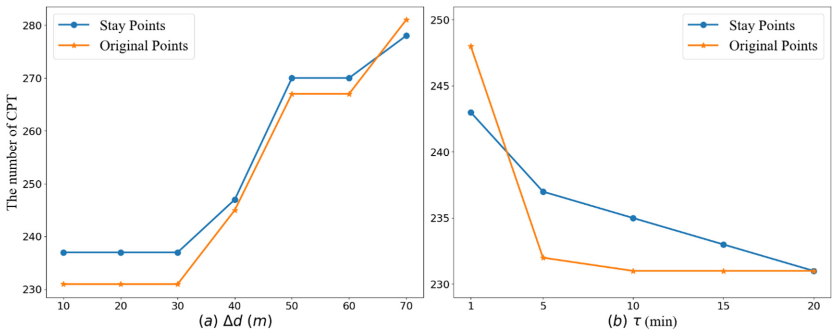

Fig. 6. The number of CPT on different parameter

5 Related Work

There is related work about chasing pattern discovery based on trajectory data. The trajectory data is a set of spatio-temporal sequences which chasing pattern is essentially a kind of sequential pattern or frequent pattern. Agrawal R and Srikant R first proposed sequential pattern mining in 1995 [10], and subsequent related research focused on spatial information only, rarely considering temporal information [11, 12]. Related work that considered temporal features mostly focused on the mining of periodic patterns [13, 14], which is not applicable to aperiodic chasing patterns. Some other special behavioral patterns, such as gathering, convoy, swarm, had received attention and research [15–17]. This kind of work does not involve the discovery of chasing patterns, but is helpful to the research of chasing patterns discovery in trajectory data.

Chasing pattern is a behavior pattern involving a pair of moving objects' trajectories. It has both coarse-grained and fine-grained spatio-temporal characteristics. Existing chasing pattern matching methods [4, 5] only involved fine-grained spatio-temporal features. If they are used for moving objects with a large range of activities, coarse-grained spatio-temporal features often cannot be found. In this article, stay-points-based chasing pattern discovery methods can process a larger range of trajectory data, and consider the fine-grained behavior characteristics of moving objects at the same time.

Existing stay point detection technologies aim to find frequent stay information at the same place at the same time, which is used to predict events such as refueling, taxiing, and switching classes. In the prediction of refueling events, in order to reduce the driver's waiting time when refueling, the stay point detection technology focuses on finding the stay position and time in the trajectory in order to optimize the distribution of gas stations [18]. In the prediction of taxi events, the stay point detection technology focuses on calculating the waiting time based on the arrival and departure times of the stay points [19]. In the prediction of the exchange class event, the stay point detection technology is to find a specific time period based on the change in the speed of the vehicle. So as to divide the driving trajectory of the same taxi to different drivers [7]. And other related studies are to find points of interest by the number of stay points in order to plan tourist routes [20]. The common feature of these stay point detection technologies is that the time and space range of the stay points are relatively fixed, and the stay time in the chasing pattern may be long or short, and the space range may be large or small. In this article, we design a stay point detection method that can dynamically detect the possible stay points in each trajectory by adjusting the threshold of the number of detected transitions.

6 Conclusion

This article proposes a chasing pattern discovery model based on stay points, and designs a chasing pattern discovery algorithm based on stay point detection algorithms. It can make full use of the time, space, and speed information contained in trajectory data to find chasing patterns between trajectories more effectively. The experiments verify the effectiveness and performance of the chasing pattern discovery algorithm proposed in this article. However, the parameters in the model designed in this article require a large number of experiments and data analysis to determine the appropriate values for the specific data set. The future work should use machine learning methods to obtain appropriate parameters automatically through a large amount of data training as well as abductive learning [21] methods with the stay point detection technology to find other behavior patterns in trajectory data. In addition, more practical application schemes of behavior pattern discovery methods should be discussed.

References

1. Alvares, L.O., Loy, A.M., Renso, C., Bogorny, V.: An algorithm to identify avoidance behavior in moving object trajectories. J. Braz. Comput. Soc. **17**(3), 193–203 (2011). https://doi.org/10.1007/s13173-011-0037-3
2. Zhao, X.-L., Xu, W.-X.: A clustering-based approach for discovering interesting places in a single trajectory. In: ICICTA, pp. 429–432 (2009)
3. Siqueira, F.D.L., Bogorny, V.: Discovering chasing behavior in moving object trajectories. Trans. GIS **15**(5), 667–688 (2011)
4. Kang, C.H., Hwang, J.R., Li, K.J.: Trajectory analysis for soccer players. In: ICDMW, pp. 377–381 (2006)
5. Zheng, Y., Zhou, X.: Computing with Spatial Trajectories, pp. 179–196. Springer, New York (2011). https://doi.org/10.1007/978-1-4614-1629-6
6. Alvares L. O., Bogorny V., Kuijpers B.: Towards semantic trajectory knowledge discovery. Data Min. Knowl. Discov., 1–12 (2007)
7. Zhang, D., Sun, L., Li, B.: Understanding taxi service strategies from taxi GPS traces. IEEE Trans. Intell. Transp. Syst. **16**(1), 123–135 (2015)
8. Zheng, Y., Zhang, L., Ma, Z., et al.: Recommending friends and locations based on individual location history. ACM Trans. Web **5**(1), 1–44 (2011)
9. Gade, K.: A non-singular horizontal position representation. J. Navig. **63**(3), 395–417 (2010)
10. Agrawal, R., Srikant, R.: Mining sequential patterns. In: ICDE, pp. 3–14 (1995)
11. Zhang, C., Han, J., Shou, L., et al.: Splitter: mining fine-grained sequential patterns in semantic trajectories. PVLDB **7**(9), 769–780 (2014)
12. Zhang, C., Ma, X., Zheng, Y., et al.: Assembler: efficient discovery of spatial co-evolving patterns in massive geo-sensory data. In: KDD, pp. 1415–1424 (2015)
13. Li, Z., Wang, J., Han, J.: ePeriodicity: mining event periodicity from incomplete observations. IEEE Trans. Knowl. Discov. Data Eng. **27**(5), 1219–1232 (2015)

14. Jindal, T., Giridhar, P., Tang, L.-A., et al.: Spatiotemporal periodical pattern mining in traffic data. In: SIGKDD Workshop on Urban Computing, pp. 1–8 (2013)
15. Zheng, K., Zheng, Y., Yuan, N.J., et al.: Online discovery of gathering patterns over trajectories. IEEE Trans. Knowl. Discov. Data Eng. **26**(8), 1974–1988 (2014)
16. Jeung, H., Yiu, M.L., Zhou, X., et al.: Discovery of convoys in trajectory databases. PVLDB, 1068–1080 (2008)
17. Li, Z., Ding, B., Han, J., et al.: Swarm: mining relaxed temporal moving object clusters. PVLDB, 723–734 (2010)
18. Zhang, F., Yuan, N.J., Wilkie, D., et al.: Sensing the pulse of urban refueling behavior: a perspective from taxi mobility. ACM Trans. Intell. Syst. Technol. **6**(3), 1–23 (2015)
19. Qi, G., Pan, G., Li, S., et al.: How long a passenger waits for a vacant taxi—large-scale taxi trace mining for smart cities. In: Green Computing and Communications, pp. 1029−1036 (2013)
20. Yuan, J., Zheng, Y., Xie, X., et al.: T-Drive: enhancing driving directions with taxi drivers' intelligence. IEEE Trans. Knowl. Discov. Data Eng. **25**(1), 220–232 (2013)
21. Zhou, Z.-H.: Abductive learning: towards bridging machine learning and logical reasoning. Sci. China Inf. Sci. **62**(7), 1–3 (2019)