國立交通大學

資訊科學與工程研究所

碩 士 論 文

利用移動物體軌跡中之線索藉由分群及
彙整技術探勘物體之移動模式

CACT : Clustering and Aggregating Clues of Trajectories for Trajectory Patterns

研 究 生：黃琬婷

指導教授：彭文志　教授

中 華 民 國　九 十 八 年 七 月

利用移動物體軌跡中之線索藉由分群及彙整技術探勘物體之移動模式

CACT : Clustering and Aggregating Clues of Trajectories for Trajectory Patterns

研 究 生：黃琬婷　　Student：Wan-Ting Huang

指導教授：彭文志　　Advisor：Wen-Chih Peng

國 立 交 通 大 學

資 訊 科 學 與 工 程 研 究 所

碩 士 論 文

A Thesis

Submitted to Institute of Computer Science and Engineering

College of Computer Science

National Chiao Tung University

in partial Fulfillment of the Requirements

for the Degree of

Master

in

Computer Science

July 2009

Hsinchu, Taiwan, Republic of China

中華民國九十八年七月

利用移動物體軌跡中之線索藉由分群及彙整技術探勘物體之移動模式

學生：黃琬婷　　指導教授：彭文志 教授

國立交通大學資訊科學與工程研究所碩士班

摘　　要

移動模式代表的是移動物體經常出現的一連串區域串列，其挑戰議題之一即是如何決定這些具代表性的區域。在真實世界中有許多原因如取樣方式、取樣頻率，機器限制，會使得軌跡資料難以描繪真實世界的移動行為。這將使得現有探勘物體之移動模式的方法難以找出準確的代表區域，同時也無法探勘出準確的移動模式。然而，即使軌跡資料只能反映出片段的移動行為，它們仍舊藏有一些關於完整之移動行為的線索。在本論文中我們提出了一個演算法ＣＡＣＴ，利用這些線索自給定的軌跡資料中探勘出移動模式。首先我們會提出對真實世界軌跡資料的觀察，由這些觀察我們設計了新的相似度來衡量兩條軌跡資料的相似程度。為了解決單一移動物體可能擁有多種移動行為的情況，我們設計了分群演算法，利用軌跡資料中的線索將代表不同移動行為的軌跡區分成不同的群組。最後我們設計了一個彙整演算法，利用軌跡資料中的線索將處在同群組中的多條片段軌跡聚合起來，重新建構成一連串具有代表性的區域串列，以描繪移動物體的移動模式。

CACT : Clustering and Aggregating Clues of Trajectories for Trajectory Patterns

Student：Wan-Ting Huang Advisor：Wen-Chih Peng

Institute of Computer Science and Engineering
National Chiao Tung University

Abstract

Nowadays, many positioning devices and techniques are more and more popular such that there are a lot of trajectories of people or vehicles can be easily obtained. From such a huge amount of trajectories collected, discovering trajectory patterns can benefit many potential and novel applications. In general, trajectory patterns indicate sequences of frequent regions that a user usually appears. One of the challenge issues in trajectory pattern mining is how to define frequent region units in trajectory patterns. In reality, there are many factors, such as sampling method, sampling frequency and device constraints, will affect the capability of original trajectory data capturing the actual movements. Thus, if the original trajectory data only coarsely capture actual movements of a user, prior works cannot accurately identify frequent regions, let alone deriving trajectory patterns. However, even if trajectories can only reflect partial movements of a user, they reveal some *clues* about the moving behaviors hidden in trajectories. Consequently, in this paper, given a set of trajectories, we propose an algorithm CACT (standing for Clustering and Aggregating Clues of Trajectories) for discovering trajectory patterns by exploiting such 'clues'. Exploiting the clues of trajectories, we first propose the similarity measurement for two trajectories by tolerating certain spatiotemporal bias. Furthermore, to deal with the existence of multiple moving behaviors in trajectories, we propose a clustering algorithm to divide trajectories with similar moving behaviors into several groups. For each group, we further propose an algorithm to derive a sequence of frequent regions with their corresponding representative line segments. To the best of our knowledge, this is the first work that claims to cluster trajectories into groups first and then derive the corresponding frequent regions within each group. Through experimental studies on both synthetic and real datasets, we show that our approach is able to capture the trajectory patterns, while handling the partial information of trajectories (i.e., the clues) and avoiding the inaccuracy problem of frequent region determination.

Keywords: trajectory, data mining, spatiotemporal

誌　　謝

首先要感謝我的指導老師—彭文志教授,在兩年的學習過程中給予我的指導與教誨。感謝我的口試委員李旺謙教授、戴碧如教授與楊得年老師，在口試的時候提出了許多寶貴的意見，讓我的論文可以做更進一步的改進。更要感謝系上辛苦的老師與系辦小姐，給予我最好的教學資源與最親切的幫助。

感謝老師的指導與教誨讓我學會了何謂研究之道,每次與老師的討論都像是戰爭一樣，新的意見被提出來、緊接著就是一連串的質疑與挑戰，只有被承認的問題才有價值，而這些問題往往會帶出更多有趣的東西。老師總是用很樂觀積極的態度帶領大家做研究，期勉我們能不斷戰勝自己持續進步，成為世界一流的人才。老師身上那用不完的活力就像新幹線的火車頭似地帶著大家向前衝，兩年的衝刺宛如經歷一場人生洗禮。感謝老師讓我不僅在研究上有了突破，精神面也能更上一層樓，面對未來的挑戰我相信我也能用這樣的精神克服難關。

感謝實驗室中的同學們,博班學長洪智傑在研究上給予了許多寶貴的意見與忠告，也扮演了實驗室的人生智者兼駐室醫生。感謝孟芬學姊關心的問候，綾音學姊珍貴的節慶應景食物，讓很少回家的我也感受得到家的感覺。感謝忠訓學長在中科計劃時的照顧,也謝謝一起做計劃奮鬥的夥伴們在我暑假去實習時幫我分擔了許多工作。感謝上一屆的碩班學長們總是很親切地提供寶貴的經驗,york、sheep、camel、講義、榕榕，真可謂是實驗室的強者傳奇，也是學弟妹們心中的榜樣。謝謝同學和學弟妹們讓實驗室充滿了歡樂，實驗室佔了碩士兩年回憶中的很大一部分，謝謝實驗室的大家讓這份回憶充滿了溫馨與歡笑。

感謝高中與大學時代的好友們，心情苦悶的時候有你們真好，特別要感謝小召學妹在四年中帶來的無窮樂趣，讓我在熬夜寫程式做研究之餘，能有個喘息休憩的空間。最後要感謝我的家人，謝謝你們兩年來的關心與支持，讓我能順利完成碩士的學業。

Contents

List of Tables

List of Figures

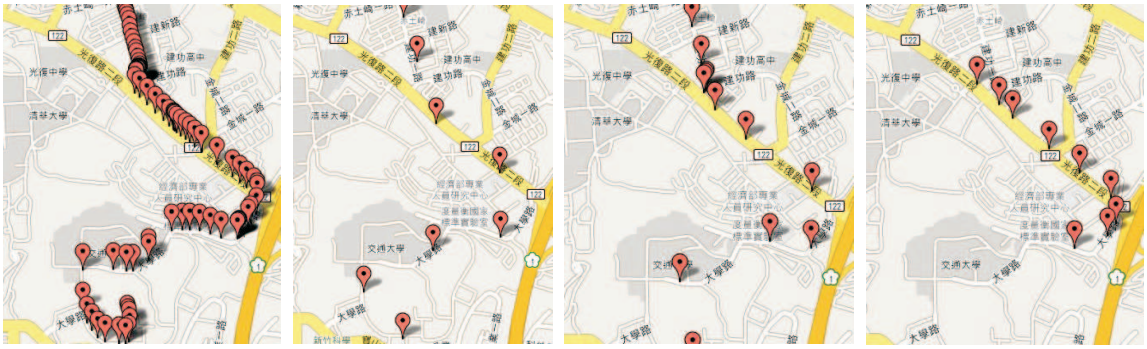
Chapter 1

Introduction

With the pervasiveness of mobile devices, the location of users is easily determined by either GPS devices or some positioning techniques. Furthermore, some softwares are able to log user movements when users go biking and traveling. Thus, a huge amount of movement trajectories are uploaded to some Web community sites [1][2][3]. From such a huge amount of trajectories collected, it is valuable to discover trajectory patterns which represent the moving behaviors hidden in trajectories. Trajectory patterns have been widely utilized in many applications such as trajectory recommendation in some trajectory sharing forums, personalized navigation and data prefetching methods in mobile computing environment.

Given a set of trajectory data, a significant amount of research efforts have proposed approaches of mining trajectory patterns. In general, trajectory patterns indicate sequences of frequent regions that a user usually appears. One of the challenge issues in trajectory pattern mining is how to define frequent region units in trajectory patterns. Previous works for determining frequent regions in trajectory patterns can be generally classified into two categories: the density-based approach and the line-based approach. In the density-based approach, a region is viewed as a frequent region if the number of trajectories passing by is larger than a pre-defined threshold. Furthermore, if nearby regions are also frequent regions, these regions could merge into one larger region. In a line-based approach, a trajectory data is first transformed as a series of line segments. If several line segments from different trajectories are close, a frequent region that contains these line segments is thus determined. The determination of frequent regions is very important since frequent regions are viewed as basic units of trajectory patterns. Without a proper determination of frequent regions, trajectory patterns are not able to capture the moving behaviors hidden in trajectories.

Clearly, the original trajectory data will have an impact on the determination of frequent regions. If the original trajectory data only coarsely capture actual movements of a user,



(a) (b) (c) (d)

Figure 1.1: Some illustrative examples extracted from Carweb datasets.

prior works mentioned above cannot accurately identify frequent regions, let alone deriving trajectory patterns. In reality, there are many factors which affects the capability of original trajectory data capturing the actual movements. To log trajectory raw data, one could set the sampling method and sampling frequency that demonstrate how and how frequent to record the location of a user, respectively. In most positioning device, there are two sampling methods: sampling by distance and sampling by time. In general, setting higher sampling frequency leads to trajectories with more fine resolution. However, setting a higher sampling frequency results in a huge amount of log data generated and the energy exhaustion of logger or GPS-enabled mobile devices. Consequently, a lower sampling frequency is likely to be set and thus trajectory data cannot reflect detailed movements of users. Figure 1.1 shows illustrative examples, where Figure 1.1(a) shows the actual movement, and Figure 1.1(b) and and Figure 1.1(c) are the trajectories following the same movement but sampling by distance and time in a lower sampling frequency, respectively. It can be seen that these trajectories are not accurately capture the actual movement in Figure 1.1(a). In addition, trajectories may be different even if a user who follows the same route. That is, GPS data points in different trajectories that demonstrate the same routing behaviors of that user may not exact the same in terms of locations and times. Figure 1.1(c) and Figure 1.1(d) show two selected trajectories with the same sampling frequency, where a real movement behavior is shown in Figure 1.1(a). Moreover, due to the natural feature of GPS or other wireless network positioning techniques, which refers to the feature that the data point determined has some tolerable errors in terms of the coordinate (i.e., location) and the time, trajectories cannot capture exact the same information of location and time even if the user follows a periodically movement.

From the observations above, prior works of generating frequent regions are not applicable since both the density-based and the line-based could not accurately determine frequent regions. For a density-based approach, it is harder to identify frequent regions that contain a

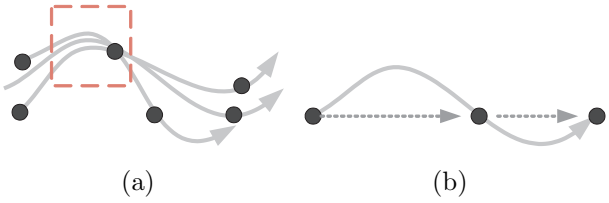


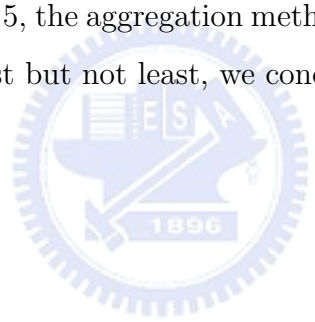
Figure 1.2: Two illustrative examples.

sufficient amount of data points. For example, the gray lines means the actual movement and the black points are sampled points. Figure 1.2(a) shows that it is possible that the frequent regions (the dashed rectangle) cannot decide accurately by the sampled points. For the line-based approach, the lines derived in a trajectory are likely not to approximate real movement paths and thus, a region that includes more close lines is hardly to derived. For example, in Figure 1.2(b), the actual movement is a S-shape curve but the line segments linked the sampled point are straight. Without a proper design of frequent regions, trajectory pattern mining cannot truly reflect frequent movement behaviors. However, even if trajectories can only reflect partial movements of a user, they reveal some *clues* about the moving behaviors hidden in trajectories. Consequently, in this paper, we propose an algorithm CACT (standing for Clustering and Aggregating Clues of Trajectories) for discovering trajectory patterns by exploiting such 'clues'. Similar to prior works in [6], trajectory patterns mined in this paper consists of sequences of frequent regions. For each frequent region, there is a representative lines which can capture geometry movements of a user within this region. Exploiting the clues of trajectories, we can distinguish whether trajectories are similar or not. Note that these trajectories may contain a variety of moving behavior of a user. Thus, it's not appropriate to put all trajectories together for the determination of frequent regions. Furthermore, to deal with the observations above, we propose a clustering algorithm to divide trajectories into several groups. Trajectories in the same group reflect the same moving behavior of a user and the number of groups is the number of moving behaviors of a user. Then, for each group, we further propose an algorithm to derive a sequence of frequent regions with their corresponding representative line segments. To the best of our knowledge, this is the first work that claims to cluster trajectories into groups first and then derive the corresponding frequent regions within each group. Because of the design, our proposed method of mining trajectory patterns is able to handle the partial information of trajectories (i.e., the clues) and avoid the inaccuracy problem of frequent region determination.

Several challenging issues arise in our proposed method, such as the formulation of similarity among trajectories, the clustering algorithm, and the derivation of frequent regions and

representative lines. Since each trajectory some clues for its actual movement, the similarity between two trajectories should be carefully designed. In light of the similarity of trajectories, we could therefore develop a clustering algorithm with the objective of extracting frequent moving behavior of a user. Clearly, the number of groups represents the number of moving behaviors of a user. According to clues of trajectories, each group should include more trajectories that are likely to have similar moving behavior so as to fully capture true moving behavior of a user. For each group derived, we propose an aggregation method to aggregate spatio-temporal information of trajectories within the same cluster and generate frequent region sequences. We evaluate our proposed algorithm in both the real dataset and the synthetic dataset. Experimental results demonstrate that our proposed algorithm is able to effectively mine trajectory patterns of a user.

The rest of the paper is organized as follows. Related works are studied in Section 2. Preliminary background is given in Section 2.1. Section 3 describes the proposed similarity for two trajectories. The clustering algorithm for trajectories with the same moving behavior is proposed in Section 4. In Section 5, the aggregation method is then presented. Experimental results are shown in Section 6. Last but not least, we conclude this paper in Section 6.3.



Chapter 2

Related Works

The problem of mining frequent moving patterns has attracted a considerable amount of research efforts. Generally speaking, the flow of mining frequent moving patterns is to first find frequent regions and then derive the relationship between these frequent regions into frequent moving patterns. According to the definition of frequent moving patterns, prior works are generally classified into two categories: spatial movement patterns and spatio-temporal movement patterns. In the first category, a frequent moving pattern refers to a sequence consisting of base station identifications or pre-defined regions. On the other hand, in the second category, frequent moving patterns are able to reflect the spatio-temporal associated relationships among base station identifications or pre-defined regions. For the first category, we mention in passing that the authors in [5] proposed an information-theoretical method to mine frequent moving patterns which are represented as a trie data structure. Moreover, the authors in [23] proposed a statistical approach to mine frequent moving patterns. In [16] and [18], the authors proposed a data mining approach for mining frequent moving patterns with the moving logs of mobile users given.

In the second category, frequent moving patterns are usually extracted from trajectories, where trajectories can reflect the actual movements. A considerable amount of research efforts have elaborated on mining spatio-temporal association rules [17][12][21][22]. In [6], the authors claimed the fuzziness of locations in patterns and developed algorithms to discover spatio-temporal sequential patterns. Furthermore, the authors in [13] proposed a clustering-based approach to discover moving regions within time intervals. In [11], the authors developed a hybrid prediction model, consisting of vector-based and pattern-based model, to predict movements of users. In [9] and [8], the authors exploited temporal annotated sequences in which sequences are associated with time information (i.e., transition times between two movements).

Prior works do not address the issue of geometric inaccuracy of trajectories. To our best knowledge, this is the first work to mine frequent moving patterns from the fragment-information trajectories. The existence of the fragment-information property brings many challenges since the geometric properties, such as angle, length and direction, cannot be used to find frequent regions directly. Different from the flow of existing works, we find the fragment-information trajectories with potentially the same moving behaviors first, and then use fragment-information trajectories in each cluster to derive a frequent region sequence. As such, our approach can not only tolerate with spatial and temporal but also overcome the geometric inaccuracy of trajectories. These features distinguish our works from others.

2.1 Preliminary

In this section, we present some assumption and notions used in this paper. Then, the problem statement is described. Finally, the overview of our proposed method is given.

2.1.1 Assumptions and Problem Statement

In this paper, we assume that the location of a user is determined by GPS devices or wireless networks. Same as in other works [6][9], a trajectory is defined as follows:

Definition 1. Trajectory representation: A trajectory T_i is a time-ordered sequence of points, denoted as $T_i =< p_{i,1} = (loc_{i,1}, t_{i,1}), p_{i,2} = (loc_{i,2}, t_{i,2}), ..., p_{i,n} = (loc_{i,n}, t_{i,n}) >$, where $t_{i,j} < t_{i,j+1}$ for all $j = 1, 2, ..., n-1$, $loc_{i,j}$ is the location at time $t_{i,j}$ and n is the length of T_i. The location determined is represented as the geometry model that consists of the latitude and the longitude of a user. Consider trajectory T_1 in Figure 3.1(a) as an example, where the black curve represents the actual movement and $T_1 =< p_{1,1}, ..., p_{1,7} >$ is a trajectory generated by GPS devices. As can be seen in Figure 3.1(a), trajectories may not always capture accurate movements of a user. Moreover, according to the setting of GPS logger, a trajectory is in fact represented partial information of a true movement path, which refers to the partial feature of a trajectory in this paper.

Same to prior works [6][9][11], trajectory patterns are sequences of regions, where regions are referred to as hot areas that a user frequently stay or pass by. As pointed out early, a grid-based approach is divided the whole space into grids and the quality of regions is mainly depended on the number of grids. Furthermore, grids may not true capture the movements of a user if the user usually appears in the boundary of grids. Thus, in this paper, we adopt a line-based approach to determine regions for trajectory patterns. Similar to the work in [6],

the region is defined as follows:

Definition 2: Frequent region Given a set of points and a central line L, the region R_L is called a frequent region if the distance between each point p and L is smaller than ϵ and the number of trajectories that cross over this region is larger than *min_sup*.

In light of the definition of frequent regions, we could therefore give the definition of trajectory patterns mined in this paper.

Definition 3: Trajectory pattern A trajectory pattern considered in this paper is a sequence of frequent regions that contain data points from at least *min_sup* trajectories (referred to relevant trajectories) and each point in relevant trajectories is mapped to one frequent region such that the distance between this data point to the central line of the frequent region is smaller than ϵ.

In this paper, given a set of trajectories, we intend to derive a set of trajectory patterns. There are some important observations of trajectories, which provides us some real phenomenon of trajectories, in the real dataset CarWeb[15].

Observation 1: For the same moving behavior, trajectories may have some data bias from the spatial and temporal (i.e., time) perspective. In our collected real trajectories, a user follows the same route to his office. These trajectories are not exact the same in their data points of trajectories. There are some bias in the spatial domain and the time domain. For example, one day, the user is little late to his office and the data points are shifted in the temporal domain. Furthermore, data points in the same location (i.e., one road segment) do not have exact the same location and time information.

Observation 2: Though setting a smaller frequency in our real dataset, the straight line with their two end points as data points in a trajectory is not usually the true movement. This is due to the road networks and the driving speed of a user. For example, if the driving speed of a user is high, the distance between two points in a trajectory is far away and thus the line between these two points cannot accurately approximate the true user movement.

Prior works in [6] that transform the original trajectory into a series of line segments are not applicable to capture the real movements of a user since lines between two consecutive points are not always movement segments of a user. According to the observations above, each data point in a trajectory is viewed as a sampling point of a true user movement. Clearly, if two trajectories have the same sampling frequency to sample the same true movement, each data point from two trajectories is possibly not the same. Since each trajectory is viewed as a sampling from a true movement, with more trajectories for the same true movement, one still could capture true movement of a user. This is due to that data points from different

trajectories that sample the same movement path, still have more possibilities to fall into some spatial areas along with the true movement paths. Furthermore, a user follows several regular true movement paths. For example, a user may have more than one working place and thus he may have more than one regular routings to his working places. Our proposed algorithm can not only mine trajectory patterns with some bias in trajectories data but also discover a set of trajectory patterns that reflect several moving behaviors of a user.

2.1.2 Overview of Our Proposed Algorithm

As mentioned above, trajectories contain partial information of true user movements and data points in trajectories are affected by some bias factors, such as the sampling frequency, positioning delay and time shifts in user movement behavior. Thus, in this paper, we propose an algorithm CACT (standing for Clustering and Aggregating Clues of Trajectories) for discovering trajectory patterns, where clues of trajectories are referred to partial information captured by trajectories. Explicitly, algorithms CACT consists of three phases: In phase 1, we formulate the similarity measurement between trajectories and our similarity formulation will take spatial and temporal bias into consideration. In phase 2, trajectories are clustered into a set of group and trajectories in the same group have similar moving behavior. Furthermore, since trajectories only consider some clues, we should carefully design the clustering method to include trajectories that demonstrate the same moving behavior. In step 3, for each group, we further aggregate trajectories in the same group for deriving a sequence of frequent regions with their corresponding central lines. In the following section, we will describe each step in detail.

Chapter 3

Similarity Measurement

In this section, we derive the similarity measurement for trajectories, which can capture the closeness of trajectories by the clues hidden in trajectories while spatial and temporal bias are taken into account. At first, we define point-to-point similarity to evaluate spatial closeness of two points by considering spatial-bias threshold ϵ. Based on point-to-point similarity, a point-to-trajectory is then defined to find the most near point in a trajectory for the given point by considering the temporal-bias threshold τ. According to the point-to-trajectory, the trajectory-to-trajectory is then defined to evaluate how closeness a trajectory to the other one is by taking both spatial and temporal bias into account. Due to the asymmetric property of the trajectory-to-trajectory similarity, *similar* and *close* relations are used to represent that two trajectories behave like each other and just one trajectory behaves like to the other one, respectively. At last, a *SC-graph* is constructed for representing these two relations among trajectories.

Each trajectory can be viewed as a time series. Therefore, one may use some existing distance measurements for time series to evaluate how similar two trajectories are, such as p-norms, dynamic time wrapping [24][14], and edit-distance-based approaches [4][7]. However, they are hardly applicable these distance measurements to evaluate how similar two trajectories are. Specifically, it is hard to compute distance between two trajectories by the Euclidean distance (p-norm measurement with p=2), because two trajectories may not have points at the same time unit. Dynamic time wrapping requires each point in a time series to match with the closet point in the other time series such that it cannot tolerate noise and capture the local similar parts between trajectories. Thus, the distance value derived by dynamic time wrapping may significantly increase if the time interval of consequent points in two trajectories are interleaving, or there exists some noises in trajectories. On the other hand, rather than matching all points between two trajectories, the edit-distance-based approach is to match

two time series by allowing some points to be unmatched. Since the length of trajectories with similar moving behavior may vary, the distance value is also affected significantly by the length such that it is hardly used to evaluate how many similar parts between two trajectories. Some similarity measurements are designed which also considers the spatial and temporal biases among trajectories [20]. However, they are usually required that there should be points at every time units in each trajectory. If the requirement above cannot be satisfied, interpolation-like approaches may be used to compensate some points for some time units. However, the trajectory may not have points in any time unit. Moreover, compensating points by interpolation cannot reflect the actual movement precisely. The reasons above motivates us to develop a new similarity measurements for trajectories.

To define the similarity for trajectories, we start from the point-to-point similarity:

Definition 2. Point-to-Point Similarity: Given a spatial-bias threshold ϵ, and two points $p_{i,\ell} = (loc_{i,\ell}, t_{i,\ell})$ and $p_{j,k} = (loc_{j,k}, t_{j,k})$, the point-to-point similarity is defined as $S_{PP}(p_{i,\ell}, p_{j,k}) = 1 - \frac{dist(p_{i,\ell}, p_{j,k})}{\epsilon}$ where $dist(\cdot)$ is the distance between locations of two points.

To tolerate some spatial bias, a parameter ϵ is used. The value of point-to-point similarity linearly decays from 1 to 0 by the distance between locations of two points. The closer the two points, the larger the value is. Once the location of two points are exactly the same, the value is 1. On the other hand, once the distance between two points are far from ϵ, the value is 0. For example, consider Figure 3.1(a) and let $\epsilon = 10$ and Euclidean distance as the distance function, it can be seen that $p_{1,3}$ is more closer to $p_{2,2}$ than $p_{2,3}$ such that $S_{PP}(p_{1,3}, p_{2,2}) = 1 - \frac{\sqrt{2}}{10} = 0.86 > S_{PP}(p_{1,3}, p_{2,3}) = 1 - \frac{2}{10} = 0.8$.

Definition 3. Point-to-Trajectory Similarity: Given a point $p_{i,\ell}$, a trajectory T_j and a temporal-bias threshold τ, the point-to-trajectory similarity is defined as $S_{PT}(p_{i,\ell}, T_j) = \max\{S_{PP}(p_{i,\ell}, p_{j,k}) | p_{j,k} \in T_j \text{ and } t_{j,k} \in [t_{i,\ell} - \tau, t_{i,\ell} + \tau]\}$.

The idea of point-to-trajectory similarity is to find the nearest point in T_j which time are allowed within τ time units from $p_{i,\ell}$. To facilitate to describe, such a closest point from $p_{i,\ell}$ is called the mapped point of $p_{i,\ell}$. In practice, there usually exists some temporal bias between two trajectories even if they follow the same moving behavior. Therefore, τ is introduced to tolerate such temporal bias for fining the mapped point of $p_{i,\ell}$. For example, suppose that we can tolerate the temporal bias to be 3 time units (i.e., $\tau = 3$), to evaluate the point-to-trajectory similarity between $p_{1,2}$ and T_2, three points $p_{2,1}$, $p_{2,2}$, and $p_{2,3}$ are considered since their time are between $t_{1,2} - 3$ to $t_{1,2} + 3$ (i.e., 1 to 7). Since $S_{PP}(p_{1,2}, p_{2,2}) = 0.8$ which owns the largest value among that from $p_{1,2}$ to $p_{2,1}$ and $p_{2,3}$, the point-to-trajectory similarity $S_{P,T}(p_{1,2}, T_2)$ is 0.8.

According to the point-to-trajectory similarity, we can further define the similarity between two trajectories:

Definition 4. Trajectory-to-Trajectory Similarity: Given two trajectories T_i and T_j, the trajectory-to-trajectory similarity is defined as $S_{TT}(T_i, T_j) = \frac{\sum_{p_{i,\ell} \in T_i} S_{PT}(p_{i,\ell}, T_j)}{|T_i|}$.

The trajectory-to-trajectory similarity is used to evaluate how a trajectory T_i is similar to the other one T_j. The value of $S_{TT}(T_i, T_j)$ is in the interval $[0, 1]$ and determined by the average of S_{PT} between each point in T_i and its mapped point in T_j with respect to the length of T_i. Since S_{TT} is derived from S_{PT} and S_{PP}, S_{TT} also takes the spatial and temporal biases into account. For example, let $\epsilon = 10$ and $\tau = 3$. In Figure 3.1(a), the arrows show the mapping from each point in T_1 to its mapped point in T_2. Therefore, we can obtain that $S_{TT}(T_1, T_2) = \{S_{PT}(p_{1,1}, p_{2,1}) + S_{PT}(p_{1,2}, p_{2,2}) + S_{PT}(p_{1,3}, p_{2,2}) + S_{PT}(p_{1,4}, p_{2,3}) + S_{PT}(p_{1,5}, p_{2,4}) + S_{PT}(p_{1,6}, p_{2,4})\}/7 = 0.64$.

There are many existing distance or similarity measurement for two time series. However, there are several important properties of S_{TT}, which makes S_{TT} more suitable to measure the similarity between two trajectories. First, S_{TT} allows the partial mapping for a trajectory to the other one. That is, a point in a trajectory does not necessary map to a point in the other trajectory. Exploiting such a feature can guarantee each point able to find its mapped point under the spatial-bias and temporal-bias constraint. For example, in Figure 3.1(a), $p_{1,7}$ has no mapped point in T_2 since there is no points which time is in $[16 - 3, 16 + 3]$. Moreover, the location of points in two trajectories may vary even if their moving behaviors are the same. Figure 3.2(b) shows an illustrative example of this case. Through this feature, S_{TT} can identify only the parts which parts of moving behaviors are likely the same between two trajectories but ignore that are not the same. Second, S_{TT} is noise tolerant. This is a crucial feature because the location data of a trajectory is inherently inaccurate. With the property of allowing the partial mapping, the value of S_{TT} does not affect significantly by noises because the noise point will has less probability to be mapped. Third, S_{TT} is asymmetric. That is, $S_{TT}(T_i, T_j)$ may not equal to $S_{TT}(T_j, T_i)$. For example, Figure 3.1(b) shows the mapping from T_2 to T_1 and we can obtain that $S_{TT}(T_2, T_1) = 0.77$ which does not equal to $S_{TT}(T_1, T_2) = 0.64$. The asymmetry is a very crucial feature since we can use it to distinguish whether a trajectory should be compensated by others or compensate others. The detail will be described later.

From the observation of the asymmetry of S_{TT}, we can define *similar* and *close* relations for any two trajectories.

Definition 5. Similar: Given a threshold λ, a trajectory T_i is *similar* to the other T_j, denoted as $T_i \sim T_j$, if and only if $\min(S_{TT}(T_i, T_j), S_{TT}(T_j, T_i)) \geq \lambda$. In brief, the *similar score*

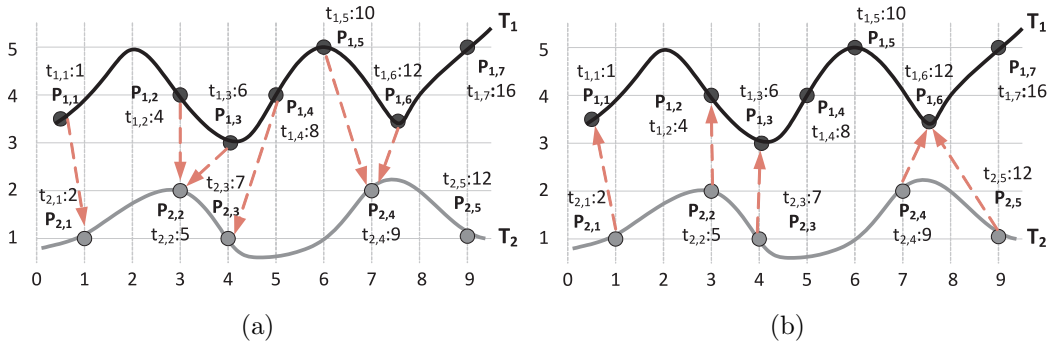


Figure 3.1: Two illustrative examples for $S_{T,T}$.

of T_i and Tj, $SS(i,j)$, is referred to the value of $\min(S_{TT}(T_i,T_j), S_{TT}(T_j,T_i))$.

Definition 6. Close: Given a threshold μ, a trajectory T_i is *close* to a trajectory T_j, denoted as $T_i \rightarrow T_j$, if and only if $S_{TT}(T_i,T_j) \geq \mu$. In brief, the *close score* from T_i to Tj, $CS(i,j)$, is referred to the value of $S_{TT}(T_i,T_j)$.

Conceptually, the *similar* relation represents that T_i and T_j intend to behave like each other. That is, they have enough amount of points which are spatially and temporally nearby to each other so that even the minimum value of S_{TT} between them is larger than a given threshold. Moreover, the similar relation is symmetric such that $T_i \sim T_j$ implies $T_j \sim T_i$. On the other hand, the *close* relation only requires that one trajectory behaves like the other one. Thus, the similar relation is not symmetric such that $T_i \rightarrow T_j$ does not necessarily imply $T_j \rightarrow T_i$. Generally speaking, we assume that two similar trajectories implies that they have close relations between them. In order to achieve the goal, we set a larger value for λ than μ. Formally, when $\lambda \geq \mu$, we can get that $T_i \rightarrow T_j$ and $T_j \rightarrow T_i$ if $T_i \sim T_j$. As such, in the following discussion, the value of λ is set to be greater than the value of μ.

Through discovering relations between two trajectories, we are able to distinguish whether a trajectory should compensate its information to others or be compensated by others based on the following observations:

Observations 1 If $T_i \sim T_j$, T_i and T_j tend to have the similar length and each point tends to be able to find its mapped point.

Observation 2: If $T_i \rightarrow T_j$, T_i tends to be shorter than T_j and only partial points of T_j will be mapped from points of T_i.

Figure 3.2 shows two illustrative example for these two observations. In this example, T_1 and T_3 follow the same movement in the black curve. Similarly, T_2 and T_4 follow the same movement in the gray curve. Let $\epsilon = 10$, $\tau = 3$, $\lambda = 0.9$ and $\mu = 0.85$. In Figure 3.2(a), it can be derived that $S_{TT}(T_2,T_4) = 0.92$ and $S_{TT}(T_4,T_2) = 0.92$. Thus, $T_2 \sim T_4$. It can

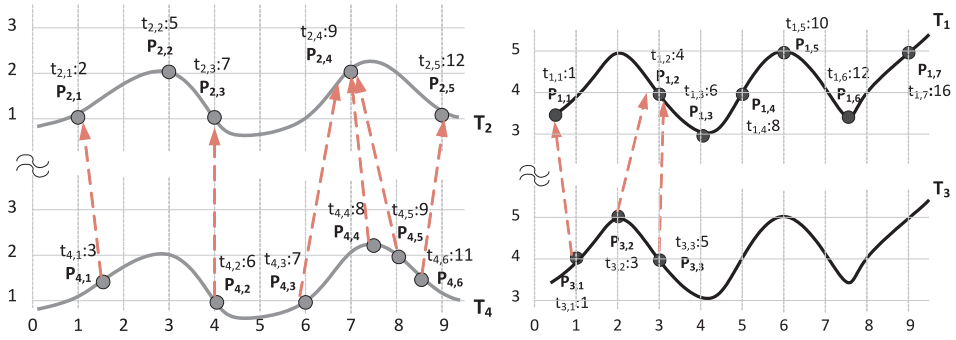


Figure 3.2: Two illustrative examples for observations.

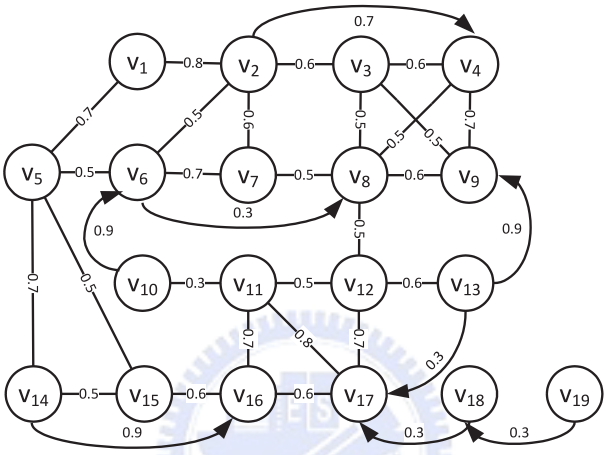


Figure 3.3: An illustrative example of a SC-graph.

be observed that the length of T_2 and T_4 are likely the same and only $p_{2,2}$ is not mapped when computing $S_{TT}(T_4, T_2)$. Based on this observation, we can conclude that a set of the trajectories which are mutually similar represent potentially the same moving behavior. On the other hand, Figure 3.2(b) shows the opposite case from the former one. In the similar fashion, we can obtain that $S_{TT}(T_1, T_3) = 0.52$ and $S_{TT}(T_3, T_1) = 0.93$. Therefore, $T_3 \rightarrow T_1$. It can be seen that T_3 is shorter than T_1 and only three points in T_1 are mapped. In this case, it can be observed that through adding three points in T_3, T_1 can be compensated and describe the moving behavior between $p_{1,1}$ to $p_{1,2}$ more precisely. Overall, the *close* relation is helpful for identifying which trajectories can compensate others.

The similar and close relations can describe the different functionalities when mining trajectory patterns. Exploiting such relations, we can identify whether a trajectory should be compensated by others or compensate others. Therefore, a SC-graph is constructed to represent the both relations among trajectories:

Definition 7. SC-Graph: Given a set of trajectories $T = \{T_1, T_2, ..., T_n\}$, a SC-graph is a weighted graph $G = (V, E_S \cup E_C)$ where $V = \{v_1, v_2, ..., v_n\}$, $(v_i, v_j) \in E_S$ with its

weight $w_S(v_i, v_j) = \min(S_{TT}(T_i, T_j), S_{TT}(T_j, T_i))$ if $T_i \sim T_j$, and $(v_i, v_j) \in E_C$ with its weight $w_C(v_i, v_j) = S_{TT}(T_i, T_j)$ if $T_i \rightarrow T_j$ and $T_i \nsim T_j$.

In a SC-graph, each vertex represents a trajectory. Once two trajectories are similar to each other, an undirected edge is then constructed between them. On the other hand, a directed edge will be constructed if one trajectory is close to the other one. Since two similar trajectories are required to be close to each other, a directed edge only exists when a trajectory is close to the other but they are not similar to each other. For example, let $\lambda = 0.5$ and $\mu = 0.3$, Figure 3.3 shows an illustrative example of a SC-graph. To facilitate to describe, a vertex in a SC-graph is equivalent to a trajectory, an edge in E_S is called a similar edge, and an edge in E_C is called a close edge in the rest of this paper.

Chapter 4

Clustering Trajectories

Since the given trajectories may contain more than one moving behavior, it is required to distinguish the trajectories with the same moving behaviors and group them into clusters. In this section, we describe our approach for grouping trajectories with similar moving behavior into clusters.

In Section 3, we introduce a SC-graph to present similar and close relations between trajectories. As such, the similar and close edges in a SC-graph represent some clues that indicates whether two trajectories represent the similar moving behavior or not. Thus, clustering trajectories with the similar moving behavior can be viewed as the procedure of exploiting some clues to clustering vertices in a SC-graph. To realize this idea, some definitions are elaborated in the following.

Definition 9. Core: Given a SC-graph $G = (V, E_S \cup E_C)$ and a threshold δ, a vertex $u \in V$ is a *core* if there exists a set of vertices C_u such that 1. for $v \in C_u$ and $v \neq u$, $(u, v) \in E_S$, and 2. for all $v, w \in C_u$, $(v, w) \in E_S$, and 3. $|C_u| \geq \delta$.

A core u in a SC-graph is a vertex with sufficient trajectories similar to it and these trajectories are mutually similar. Thus, a core set C_u contains trajectories which can most likely represent the same moving behavior. The value of δ is usually set to be at least 2 since the moving behavior described by a trajectory which is not similar to anyone is not enough confident. For example, let $\delta = 2$, v_2 is a core and $C_{v_2} = \{v_2, v_6, v_7\}$ where v_6 and v_7 are the neighboring vertices of v_2 in E_S, each vertex has similar edge to others, and $|C_{v_2}| = 3 \geq \delta = 2$.

Even if some trajectorie follows the same moving behavior, it is possible that they are not in the same core due to the natural of trajectories. However, some 'clues' may exist to indicates two cores with the similar moving behavior. This definition is elaborated in the following.

Definition 10. Directly Clue-Reachable: A vertex u is directly clue-reachable to a

vertex v, denoting as $u \rightsquigarrow v$, if v is a core and u is adjacent to v in E_S or E_C.

Directly clue-reachability shows a vertex u with the same moving behavior to a core. A vertex u can show it following the same moving behavior of a core through a similar or a close edge, respectively. Obviously, all vertices in C_v are mutually directly clue-reachable. For example, $v_5 \rightsquigarrow v_6$ since v_6 is a core and $(v_5, v_6) \in E_S$; $v_1 \rightsquigarrow v_2$ since v_2 is a core and $(v_1, v_2) \in E_C$.

Through the directly clue-reachability, we can find those vertices which potentially represent the similar moving behavior as a core. In the following definition, we extent the directly clue-reachability to clue-reachability which can describe a vertex following the similar moving behavior through many clues indirectly.

Definition 11. Clue-Reachable: A vertex u is clue-reachable to a vertex v, denoting as $u \rightsquigarrow^* v$, if there exists a chain of vertices $v = v_1, v_2, ..., v_n = u$ such that $v_i \rightsquigarrow v_{i+1}$ for all $i = 1, 2, ..., n-1$.

For example, $v_5 \rightsquigarrow^* v_8$ through the path $v_5 \rightsquigarrow v_6 \rightsquigarrow v_7 \rightsquigarrow v_8$.

Based on clue-reachability, we can further define the clue-connection from one core to the other core as follows:

Definition 12. Clue-Connect: A core u is clue-connected to v if there exists a core w such that $x \rightsquigarrow^* y$ for all $x \in C_u$ and for some $y \in C_w$, and $y' \rightsquigarrow^* z$ for all $y' \in C_w$ and for some $z \in C_v$.

Conceptually, through clue-connection, we can imply the moving behavior of a core u is similar to that of a core v. To ensure sufficient clues to support that, each vertex in C_u should be clue-reachable to some vertices of an intermediate core sets. That is, all vertices in C_u, i.e., trajectories stating the similar moving behavior of u, should follow the similar moving behavior as an intermediate core set. Similarly, all vertices in this intermediate core sets should follows the same moving behavior as the core v. For example, v_{11} is clue-connected to v_8. It can be seen that there is a core v_5 such that all vertices in $C_{v_{11}}$ are clue-reachable to some vertices in C_{v_5} (i.e., $v_{11} \rightsquigarrow v_5$ and $v_{10} \rightsquigarrow v_5$), and all vertices in C_{v_5} are clue-reachable to some vertices in C_{v_8} (i.e., $v_1 \rightsquigarrow^* v_3$ and $v_5 \rightsquigarrow^* v_8$).

For a core, there may be several cores clue-connected to it. To ensure that trajectories with the most similar moving behavior are grouped into a cluster, we derive a measurement *clue-gain* to evaluate how much 'clue' a core set can obtain via the other one.

Definition 13. Clue-Gain: Consider two sets S and T. Let E_S^{st} and E_C^{st} be the set of similar and close edges from S to T, respectively. The clue-gain $ClueGain(S,T) = \alpha \times |E_S^{st}| \times \sum_{e \in E_S^{st}} w_S(e) + \beta \times |E_C^{st}| \times \sum_{e \in E_C^{st}} w_C(e)$.

Generally speaking, more similar/close edges from S to T implies that S is more likely to represent the similar moving behavior as T. Also, the weights between these edges should be taken into account. The higher weights of these edges, the more similar the moving behaviors of S and T. Therefore, the clue-gain is proportional to the number of similar and close edges and the corresponding weights from S to T. Moreover, the similar edges should be weighted higher than the close edges because the similar edges represent that the moving behaviors of two vertices are mutually similar and the close edges only represents the moving behavior of one vertex is like to the other. Thus, two constants α and β are used for weighting the similar and close edges, respectively. Usually, the value of α should be at least two times larger than β. By the definition of similar and close scores, $SS(i,j) = \min(S_{TT}(T_i,T_j), S_{TT}(T_j,T_i)) \leq S_{TT}(T_i,T_j) = CS(i,j)$. Thus, $2SS(i,j) \leq CS(i,j) + CS(j,i)$, which shows that one similar edge is at least two times important as a close edge. Thus, α should be set two times larger than β.

For example, let $\alpha = 2$ and $\beta = 1$. $ClueGain(C_{v_{12}}, C_{v_3}) = 2 \times 1 \times 0.5 + 1 \times 1 \times 0.9 = 2.9$ and $ClueGain(C_{v_{12}}, C_{v_{11}}) = 2 \times 2 \times (0.5 + 0.7) + 1 \times 1 \times 0.3 = 5.1$. Obviously, $C_{v_{12}}$ intends to show more similar moving behavior to $C_{v_{11}}$ than C_{v_3} since there are more similar edges from $C_{v_{12}}$ to $C_{v_{11}}$ than to C_{v_3}.

According to the clue-connected and the clue-gain, we can formulate the problem of clustering trajectories with similar moving behavior as follows:

Definition 14. Cluster: A cluster C is a set of vertices satisfying the following conditions: 1. for all $u \in C$, there exists a core $v \in C$ such that u is clue-connected to v (connectivity); 2. for all core sets $C_u \in C$, the core set C_v which can induce $ClueGain(C_u, C_v)$ maximal is also in C (compactness); 3. $|C| \geq min\_sup$ (frequentness).

The first requirement states that a cluster is composed of many cores which have clues to support them describing the similar moving behavior. The second requirement describes the compactness of a cluster, where each core set should be in the same cluster with the core set which can make the clue-gain maximal. That is, each core set is used to interpret the moving behavior with the strongest clues from this core set. On the other words, a core set will not interpret the moving behavior with weaker clues. To ensure derived regions being frequent, a cluster should contain more than min_sup vertices which is describing in the third statement.

We propose a clustering algorithm to find clusters in a SC-graph. In nut shell, this algorithm first discovers all core sets, then merges them according to their clue-gains, and adds some non-cores into clusters for enriching information of clusters at last. The algorithmic form is listed in Algorithm 1.

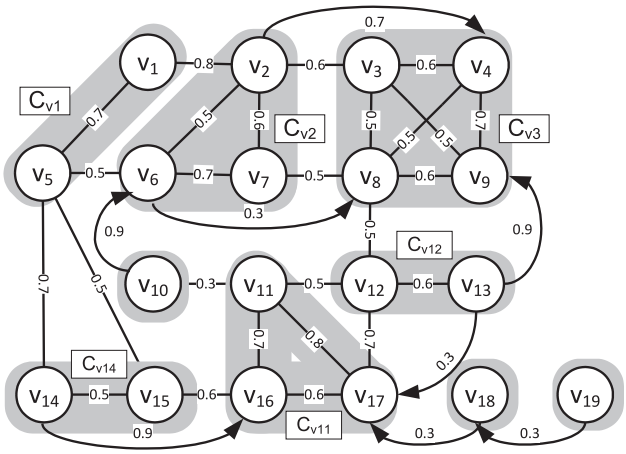


Figure 4.1: A scenario of clustering in a dual graph.

Note that a core set in a SC-graph is equivalent to a clique with size $\geq \delta$ on E_S. Thus, in the beginning, we find a clique cover on E_S, where a clique cover refers to a set of clique with their union being the whole graph. There are many existing heuristic algorithm to find a clique cover efficiently [10]. One of famous heuristic algorithms is based on greedy strategy which idea is to always select the highest degree vertex, and to pick its adjacent vertices which have edges mutually to form a clique. For example, v_8 owns the highest degree in this graph (only considering E_S). There are five vertices adjacent to it, say v_7, v_3, v_4, v_9, and v_{13}. It can be verified that only v_3, v_4, and v_9 have edges between each other. Thus, the first clique $\{v_3, v_4, v_8, v_9\}$ is then generated. Cliques of a clique cover in our example are shaded in Figure 4.1.

After finding a clique cover, we can identify those clique with size $\geq \delta$ as the core sets. For example, let $\delta = 2$, C_{v_1}, C_{v_2}, C_{v_3}, $C_{v_{11}}$, $C_{v_{12}}$, and $C_{v_{14}}$ are core sets. As long as deriving the core sets, each core set computes the clue-gain from it to all one-step clue-connected core sets. Then, a core set is merged to the core set with the maximal clue-gain. For example, for $C_{v_{12}}$, there are two one-step clue-connected core sets C_{v_3} and $C_{v_{11}}$ with the clue-gains $ClueGain(C_{v_{12}}, C_{v_3}) = 1.9$ and $ClueGain(C_{v_{12}}, C_{v_{11}}) = 5.1$, respectively. Thus, $C_{v_{12}}$ is merged with C_{v_3} rather $C_{v_{11}}$. The other example is that $C_{v_{14}}$ is merged with C_{v_1} due to $ClueGain(C_{v_{14}}, C_{v_1}) = 4.8 > ClueGain(C_{v_{14}}, C_{v_{11}}) = 2.1$. Similarly, C_{v_1} is merged into C_{v_2} and C_{v_2} is merged into C_{v_3}. Consequently, we can derive two clusters: $\{C_{v_1}, C_{v_2}, C_{v_3}, C_{v_{14}}\}$ and $\{C_{v_{11}}, C_{v_{11}}\}$. It is worth mentioning that the merged cliques are guaranteed to be clue-connected since each clique can be only merged with its one-step clue-connected clique, thereby satisfying the requirement 1 and 2 of a cluster. At last, some cliques with size $\leq \delta$ are considered to join some clusters to compensate the moving behaviors represented by this cluster. As such, a non-core joins a cluster with a core which can induce the maximal clue gain. In

this case, a non-core does not merge the other non-core because a chain of non-cores with different moving behavior may be contained in a cluster, especially for non-cores with only one vertex. For example, v_{18} and v_{19} form such a chain. It can be seen that the v_{18} is close to v_{17} which indicates it can compensate the information of $C_{v_{11}}$. However, it is not obvious whether v_{19} can be used to compensate or not. After that, clusters with less than min_sup vertices and the remaining non-cores are eliminated. Following the example, let $min\_sup = 5$. The cluster $\{C_{v_{11}}, C_{v_{11}}\}$ and the vertex v_{19} are eliminated. Consequently, there are two clusters $\{C_{v_1}, C_{v_2}, C_{v_3}, C_{v_{14}}, v_{10}\}$ and $\{C_{v_{11}}, C_{v_{12}}, v_{18}\}$ as the final results.

Algorithm 1: Clustering Trajectories

Input : A SC-graph: $G = (V, E_S \cup E_C)$
Output : A set of clusters: C

$C \leftarrow \phi$;
$K = \{K_1, K_2, ..., K_m\} \leftarrow$ a clique cover of G;
$CORE \leftarrow$ cliques in K with the size $\geq \delta$;
for *each clique* K_i *in* $CORE$ **do**
    Compute the clue-gain with all one-step clue-connected cliques in $CORE$;
    $C_i = K_i$;
    add C_i into C;
for *each clique* K_i *in* $CORE$ **do**
    $K_{max} \leftarrow$ the clique in $CORE$ which can maximize $ClueGain(K_i, K_{max})$;
    $C_{max} \leftarrow$ the cluster containing K_{max};
    $C_{join} \leftarrow$ the cluster containing K_i;
    $C_{max} \leftarrow C_{max} \cup C_{join}$;
for *each clique* K_j *in* $K - CORE$ **do**
    $K_{max} \leftarrow$ the clique in some C_{max} which can maximize $ClueGain(K_j, K_{max})$;
    Put K_j into C_{max};

Selection of Thresholds

The selection of thresholds usually depends on user's requirements and the properties of the environment. However, setting λ and μ are not straightforward tasks. The selection of thresholds highly affects the structure of a SC-graph since the number of edges significantly depend on the thresholds for similar and close relations (i.e., λ and μ). The larger λ and μ restrict whether two trajectories have similar and close relations or not more seriously. Thus, larger thresholds incur the fewer edges in a dual graph and make a dual graph more sparse. A cluster in a sparse dual graph may contain only few trajectories such that it is hard to aggregate them to obtain the frequent movement precisely. The smaller λ and μ makes a dual graph more dense. However, it is easy for a cluster to contain more irrelevant trajectories such that the frequent regions cannot be derived precisely. Therefore, the results of clustering are highly dependent on the values of λ and μ.

Here, we propose a heuristic for selecting λ and μ adaptively according to the distribution

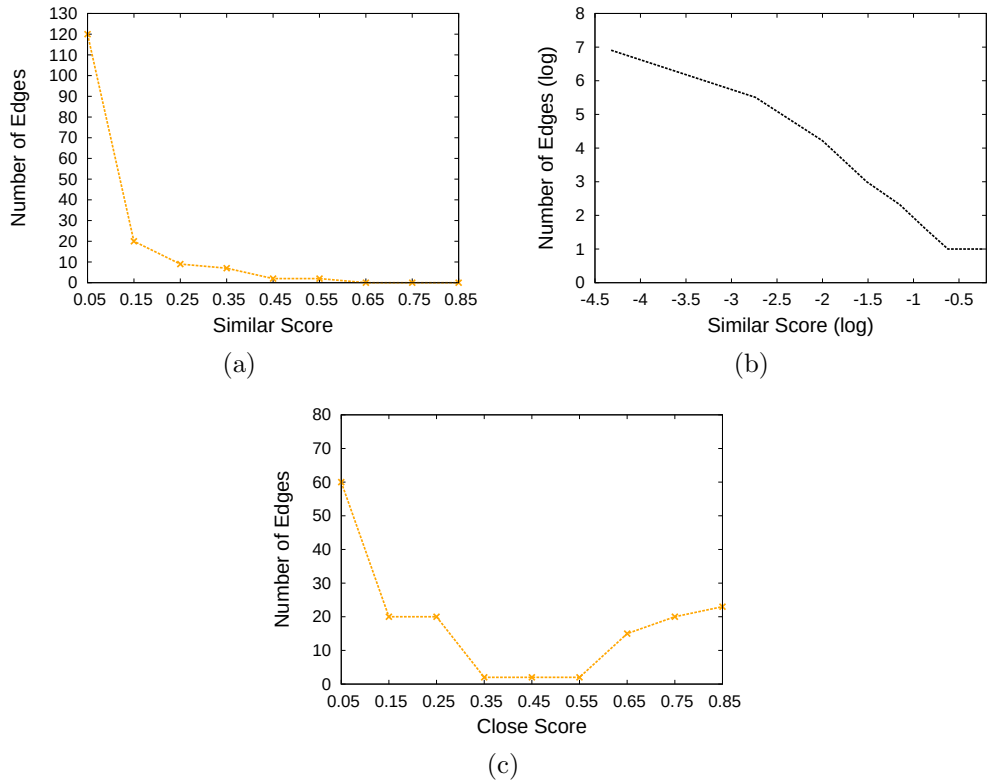


Figure 4.2: Distribution of (a) similar and (b) close scores

of similar and close scores. To determine a proper value of λ, we can observe that the distribution of similar scores. Figure 4.2 shows an experimental result for the distribution of similar scores in our real datasets. Figure 4.2(a) shows that there are a lot of edges with extremely small similar score, and relatively fewer edges for the remaining similar scores (i.e., long tail). Figure 4.2(b) shows the same plot as above one, but the same distribution shows itself to be linear on a log-log scale, which is the characteristic signature of the power law distribution. Thus, by the observations above, we can conclude that the similar scores follow the power law distribution.

As such, the threshold λ should not be these small similar score since it will make a dual graph too dense. In this case, we should not select the threshold λ to be 0.05. To prevent the threshold λ too large, a heuristic approach to select the threshold λ is to select the average similar score in the long tail. For the efficiency sake, the long tail can be simply determined by 80-20 principle where we suppose that the edges with 20% largest similar score form the long tail [19]. In our example, the edges with similar score larger than 0.15 form the long tail. On the other hand, Figure 4.2(c) shows that the distribution of the close scores tends to have two peaks. Note that the close relation is used to identify which trajectories should provide information to compensate the other one. Therefore, to prevent less trajectories with different moving behavior to put into a cluster, the edges with lower close score should not exist in

a dual graph. As such, the threshold μ can be selected to be the average of all close scores which value can keep the right peak and discard the left peak. In this example, the value of μ is set to be about 0.4. In our latter experimental results, we will show that the heuristic approach can group trajectories with the same moving behavior into the fewer clusters.

Chapter 5

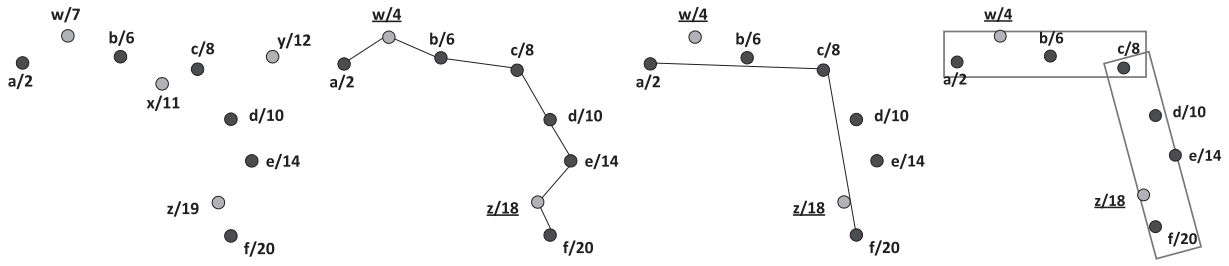
Aggregation Phase

In clustering phase, the trajectories are divided into several clusters where each cluster contains more than *min_sup* trajectories. In this phase, the spatial and temporal information of trajectories in a cluster will be aggregated and then a frequent region sequences is generated for each cluster.

The trajectories in a cluster may represent the same moving behavior. However, it is not a trivial task to aggregate the information of these trajectories because there may exist some spatial bias, temporal bias, and noise data in them. To overcome these issues, a trajectory which can best represent the moving behavior of trajectories in a cluster will be chosen. Such a trajectory is referred as to the *kernel*. The information of other trajectories are adjusted to compensate the kernel. Once obtaining the compensated kernel, the coming issue is how to decide the minimal number of regions which can satisfy the spatial bias threshold ϵ.

Since the weight of two similar edges represents how much these two trajectories are similar to each other, the vertex which the highest total weights of the similar edges incident to refers the trajectory which the most trajectories in a cluster are similar to. For example, in Figure 4.1, v_3 is the kernel in the cluster $\{C_{v_1}, C_{v_2}, C_{v_3}, C_{v_{14}}, v_{10}\}$. Note that the kernel is likely from more larger cores. A larger core has more mutually similar vertices such that each vertex has more similar edges incident to it. The total weights of a vertex in a larger core is more easily larger than that in a smaller core. In addition, we have more confident to the moving behavior describing by a larger core than a smaller one. It satisfies that we shall select the trajectory which can most represent the moving behavior of this cluster. Moreover, a larger core tends to have more close edges incident to it. It also follows the intuition that the kernel can be compensated the moving behavior from other trajectories.

Since not all information of trajectories can be used to compensate the kernel, the order of adding trajectories should be carefully decided to make as more trajectories able to compensate



(a) (b) (c) (d)

Figure 5.1: An illustrative example of our aggregation algorithm

their information to the kernel as possible. As such, the trajectories which are most likely similar to the kernel should be first considered. To evaluate how a trajectory is similar to the kernel, we should first consider the minimal steps from the trajectory to the kernel, which can be done by BFS. Once a vertex can achieve the kernel by fewer edges, this vertex has less spatial and temporal bias to the kernel with higher probability. Moreover, the path that induces the minimal steps from the trajectory to the kernel is also important. Product of the weights along the path implies that how much this trajectory is similar to the kernel transitively. Overall, the trajectory with smaller BFS steps and the higher product of weight along its BFS path should be first considered. For example, consider the cluster $\{C_{v_1}, C_{v_2}, C_{v_3}, C_{v_{14}}, v_{10}\}$ in Figure 4.1. In this cluster, the kernel is v_3. The vertices v_6 and v_7 are two-step far from the kernel. From v_6 to v_3, the maximal product of weight is $0.5 \times 0.6 = 0.3$. From v_7 to v_3, the maximal product of weight is $0.5 \times 0.5 = 0.25$. Thus, v_6 has higher priority to compensate the kernel than v_7 does.

After deciding the order of compensating the kernel, the next task is to adjust the spatial and temporal information of other trajectories such that these information can be used to compensate the kernel. The concept of our aggregation algorithm can be best understood by the example in Figure 5.1. Suppose that the black points are from the kernel and the grey points are from the compensating trajectory. The number associated with each point represents the time. In the beginning, all points of the compensating trajectory are spatially projected as shown in Figure 5.1(a). Among the compensating points, The point w has some temporal bias with the kernel because it locates between kernel points a and b but the value of time of w is not between that of a and b. The point x has such temporal bias as well. In addition, the point y is a noise point which is too far from the other points. Then, according to the points of the kernel, the temporal information of compensating points will be adjusted. Suppose that a compensating point p locates between two kernel points q and r. If p is between $[t_q - \tau, t_r + \tau]$ where τ is the temporal-bias threshold, then its time is adjusted by the proportion of its distance to q and r Specifically, $t_p = t_q + (t_r - t_q) \times \frac{dist(p,q)}{dist(p,q)+dist(p,r)}$. Otherwise,

the point is discarded. Such adjustment is reasonable because a temporal-bias τ is allowed when computing the similarity between two trajectories. For example, let $\tau = 2$. Suppose that the distance between a and w equals to that between b and w. Since the time of w is 7 which is between $[2-2, 6+2]$, the time of w is adjusted as $2 + (6-2) \times \frac{dist(a,w)}{dist(a,w)+dist(w,b)} = 4$. On the other hand, the time of x is 11 which is outside the interval $[6-2, 8+2]$. Thus, the compensating point x is discarded. In the similar fashion, the time of z is adjusted to 18. Next, the noise points are discarded. Following the notations above, the point p is a noise point if $dist(p, \overline{qr}) > \epsilon$. The point y is the noise point and thus eliminated. Figure 5.1(b) shows the results after adjusting the temporal information and eliminating noise points. The procedure repeats until all compensating trajectories are added.

Algorithm 2: Aggregation Algorithm

Input : A set of clusters: C
Output : A set of frequent region sequences: R

for *each cluster* $K \in C$ **do**
  $T_{ker} \leftarrow$ the kernel trajectory of C;
  for *each vertex* $v \in K$ **do** Compute its BFS steps and largest weight product to the kernel;
  for *each trajectories* T *in the descending order of BFS steps and weight products* **do**
  Spatial projection all points of T;
  for *each points* p **do**
    $q, r \leftarrow$ two points in the kernel that p locates between them;
    if $t_p \in [t_q - \tau, t_r + \tau]$ *and* $dist(p, \overline{qr}) \leq \epsilon$ **then**
      $t_p = t_q + (t_r - t_q) \times \frac{dist(p,q)}{dist(p,q)+dist(p,r)}$
    else
      discard p;
    end
  end
  $L \leftarrow$ lines obtaining by Douglas-Peucker algorithm;
  $\Omega \leftarrow$ regions by central lines L;
  Add Ω into R;
end

After adding all compensating points, Douglas-Peucker algorithm are used to determine the number of regions. The purpose of this algorithm is that finding the minimal line segments which the distance of each point to the corresponding line is smaller than a threshold ϵ. Therefore, the minimal number of regions can be obtained while the distance between each point to the line can be guaranteed to be smaller than a threshold ϵ. Conceptually, the algorithm recursively divides the line. Initially a line segment with the first and the last points is constructed. If the farthest point to the line segment is closer than ϵ, it represents the point can be represented by this line. Otherwise, if the point furthest from the line segment is greater than ϵ, the original line segment will be separated into two line segments at this point. The algorithm recursively calls itself until the distance of all points to the derived lines are smaller than ϵ. Taking the line segments in Figure 5.1(b) as input, Figure 5.1(c) shows the

final results where two line segments are derived. Consequently, by viewing the derived lines as the central lines, the regions can be easily derived. The final results are shown in Figure 5.1(d).

Chapter 6

Performance Evaluation

In this section, the effectiveness and efficiency of mining trajectory patterns from trajectories are evaluated. In Section 6.1, we present the environments and settings in our experiments. All experiments are conducted by both the synthesis dataset and the real dataset. The comparison between our approach and the existing work are shown in Section 6.2. Sensitivity analysis in several parameters are also investigated in Section 6.2.1.

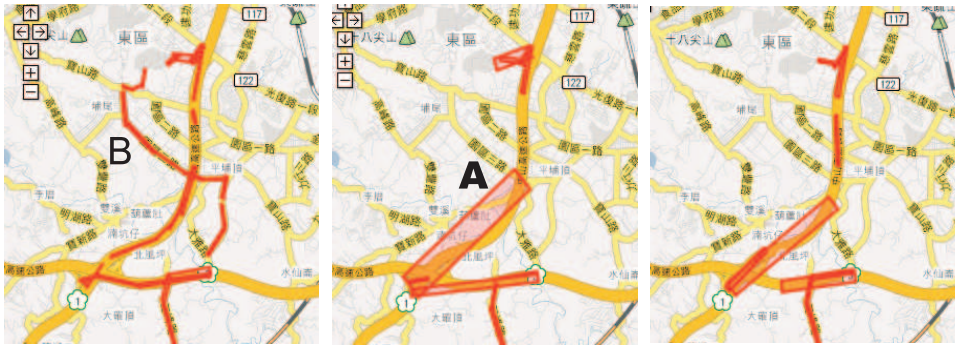
6.1 Experimental Environment

In our experiments, both real and synthesis datasets are used to evaluate the existing works and the proposed methods. For real datasets, we extract trajectories from a GPS-based testbed, *CarWeb*, which aims at collecting real trajectories of users [15]. In *CarWeb* system, each user can obtain his location from GPS every five seconds and upload his location to the *CarWeb* server. Note that, we can manually category the trajectories in *CarWeb* dataset such that we could have the ground truth about the moving behaviors represented by trajectories. In our experiments, we choose three kinds of trajectories which represents 3 kinds of frequent moving behaviors and one kind of trajectories which represents infrequent ones. Specifically, there are 30, 20, and 10 trajectories with 300, 130, and 500 points in average, respectively. There are 3 infrequent trajectories with 160 points in average. On the other hand, for synthetic datasets, we construct a simulator to generate many synthetic trajectories by given source trajectories as inputs. The source trajectories we extract are generated in a very high sampling rate and we also manually adjust them to ensure the correctness. Each synthetic trajectory is a variant from a input trajectory, where the time of each point may shift by t time units and the location of each point may shift by the angle θ and the distance r in a probability p_{bias}. The value of θ and r are uniformly distributed from 0 to 2π and 0 to ϵ (i.e., the threshold

for spatial bias), respectively. In our simulation, there are four kinds of source trajectories, which have 3000, 4000, 5000, and 6000 points, respectively. To simulate the different sampled rates, we induce the loss rate P_{loss} to determine the probability of a point in a highly-sampled trajectory will be discarded. Explicitly, each point in a trajectory is discarded with probability P_{loss}. Thus, a trajectory tends to be more inaccurate from the real movement under a higher loss rate. The default value of parameters are listed as follows: $\tau = 30$ minutes, $\epsilon = 10$ meters, $min\_sup = 0.1$, the total number of trajectories is 200, $r = 5$ meters, $t = 30$ minutes, $P_{bias} = 80\%$, and $P_{loss} = 50\%$.

For the comparison purposes, the method of mining spatio-temporal frequent patterns in [6], denoted by SFP, is implemented. Since this approach is not designed for mining trajectory patterns from trajectories, we exploit linear interpolation and cubic spline interpolation to estimate the locations for low-sampled trajectories and use these compensated trajectories as inputs of SFP, say SFP-L and SFP-C. Also, since SFP cannot cluster trajectories with the different moving behavior, the inputs of SFP are the trajectories with the same moving behaviors.

To evaluate experimental results, three performance metrics, *precision*, *recall* and *execution time*, are used. Since both SFP and our methods use a sequence of spatial regions to represent the frequent patterns from trajectories, a smaller region may cover less road segments, which can describe which road segments are frequent more precisely. Since the both datasets are obtained from the movement on a road network, each data point in a trajectory can be also bind with a road segment id. Thus, a trajectory can represent as a sequence of road segments. By presenting each trajectory into a sequence of road segments, the conventional approaches for mining frequent itemsets, such as apriori, can be used to find frequent road segments from the given trajectories. The *precision* is used to evaluate how precise the derived frequent regions in different approaches. Let C be the road segments covered by the derived regions and F be the frequent road segments. The *precision* is formulated as $L(C \cap F)/(L(C \cap F) + L(C \cap \bar{F}))$, where $L(\cdot)$ represents the total length of the roads. A higher precision value means that the derived region tends to cover more frequent road segments and fewer infrequent ones. On the other hand, the *recall* is used to evaluate the area that the derived regions can cover the frequent road segments derived by apriori. The *recall* is formulated as $L(C \cap F)/(L(C \cap F) + L(\bar{C} \cap F))$. A higher recall value means that more frequent road segments can be covered by the derived region. At last, the *execution time* is used to measure the efficiency and the scalability of the proposed method.



(a) (b) (c)

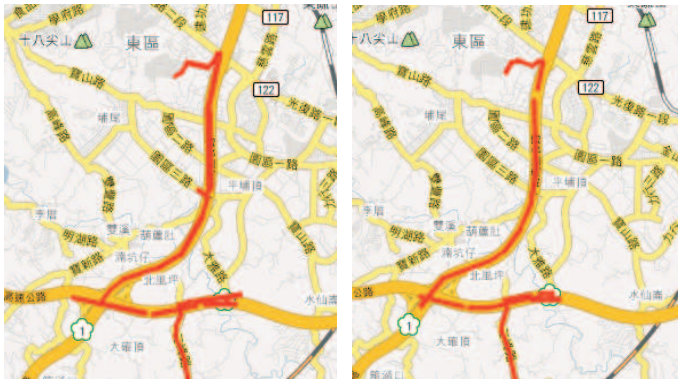
Figure 6.1: Three experimental results of an existing work.

6.2 Comparison with SFP

In this section, we evaluate our approach and SFP in terms of the visualized results, the precision and the recall.

We first show the frequent regions discovered from real datasets in a visualized manner. Figure 6.1(a) shows the result of SFP when the given trajectories are manually compensated and corrected the position of each point. The derived regions are thin and located on the road segments, which are consistent with the fact that trajectories in a road network moves on roads. Then, consider $loss_{rate} = 50\%$, the derived regions derived by SFP is shown in Figure 6.1(b). It can be seen that region A locate outside the road segments, which are inconsistent to the fact of the movement on a road network. Moreover, some regions disappear. To deal with the problem, one may propose that interpolation can be used to compensate the location data between the remaining points. Given the compensated trajectories, Figure 6.1(c) shows the derived regions. Comparing with Figure 6.1(b), such strategy is effective because more derived regions locate on the roads, especially the two regions at the center. However, it still does not work well because some frequent regions are still missing. In brief, from these experimental results, we can observe that the frequent regions cannot be derived precisely, even incorrectly, by the line-based approach.

As mentioned above, there are four kinds of moving behaviors within the given trajectories in real datasets. Figure 6.2 shows one of the frequent regions which is found by our approach. Comparing the results of SFP in Figure 6.1(a) and that of our approach in Figure 6.2(a), it can be seen that the frequent regions found by two approaches are similar, which shows our approach can also find frequent spatio-temporal regions found by SFP. Moreover, the common regions in Figure 6.2(a) and Figure 6.1 represent the most frequent road segments among trajectories. Figure 6.2(b) shows the result with $P_{loss} = 0.5$, which the derived regions



(a) (b)

Figure 6.2: Two experimental results of our approach with (a) $P_{loss} = 0$ and (b) $P_{loss} = 0.5$.

are almost the same as the case when $P_{loss} = 0$. Therefore, we can conclude that our approach is robust when P_{loss} varied such that can find frequent regions from trajectories. It is worth mentioning that our approach does not find frequent regions B found by SFP in Figure 6.1(a). This is because in the aggregation phase, we choose only one kernel trajectory which is similar to most trajectories in a cluster and only those points which comply the moving behavior with the kernel trajectory are used to compensate. As such, only the regions which present the most frequent moving behavior are derived and some regions which are less frequent will not be generated. We left the methodology of solving this issue as a future work.

We now evaluate the Precision and recall. As mentioned above, SFP takes trajectories with the same moving behavior as inputs. The shown results of SFP-L and SFP-C are the average of the precision and recall of all clusters. Figure 6.3 shows the precision and recall with p_{loss} varied in real datasets. As shown in Figure 6.3(a), our approach outperforms SFP-L and SFP-C. Moreover, the precision of our approach can keep almost constant while that of SFP-L and SFP-C decrease with P_{loss} increasing. Figure 6.3(b) shows the recall of three approaches. Although the recall of all approaches decrease with P_{loss} increasing, the recall of our approach is much higher than other two approaches. Since the regions tend to be boarder with a larger P_{loss}, more infrequent road segments are included into the derived regions such that the recall decreases. Figure 6.4 shows the results for the synthesis datasets. Since the trajectories in synthesis datasets are more complicated, which have more turns and varied length, than that in real datasets, the precision and recall are about 20% lower than that in real datasets. However, the similar results can be obtained as the results in real datasets.

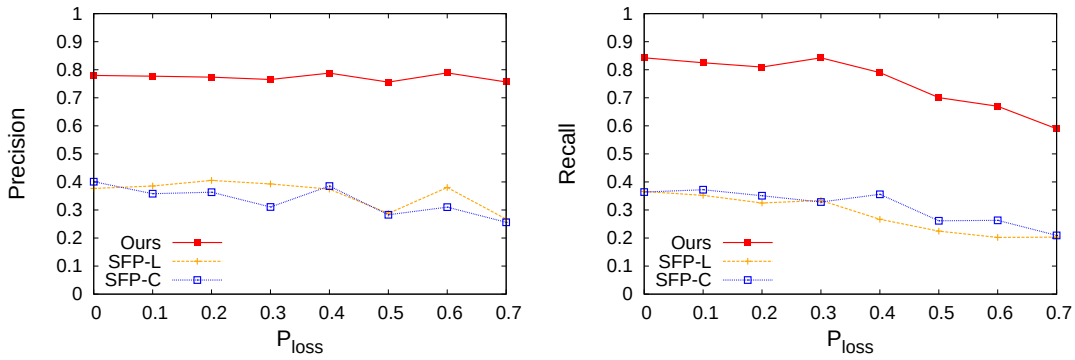


Figure 6.3: The impact of Precision and Recall in real datasets with P_{loss} varied.

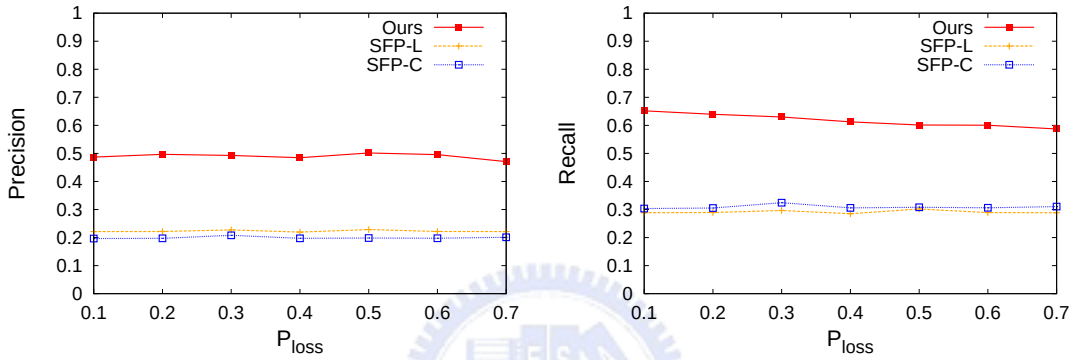


Figure 6.4: The impact of Precision and Recall in synthesis data sets with P_{loss} varied.

6.2.1 Sensitivity Analysis

In this section, we further investigate the parameters used in our algorithm. First, the effectiveness of algorithm ClusDG is presented. Then, we examine the impact of thresholds and the environment settings for the mining results. At last, the scalability of our approach is discussed.

6.2.2 The Heuristic of Threshold Selection

To construct the dual graph, there are two thresholds λ and μ to be decided. A heuristic approach is proposed in the former section. This heuristic approach is based on the observation that the distribution of weight of edges follow the power law distribution. We first vary the number of each kind of trajectories and observe the distribution of the weight of edges. Figure 6.5(a) shows that the weight of edges follows the power law distribution with different parameters. With the increasing of trajectories, the edges with lower similar scores increase such that the long tail of the power law distribution becomes shorter. In Figure 6.5(b), we

Number of Trajectories	Range of λ	Range of μ	Heuristic λ	Heuristic μ
80	.01∼.02	.016∼.02	.016	.015
120	.01∼.025	.015∼.015	.019	.018
160	.02∼.045	.01∼.037	.024	.021

Table 6.1: Comparison for optimal and heuristic solutions.

fix the number of trajectories and vary the loss rate P_{loss}. It can be seen that the pow law distribution is little sensitive with P_{loss}. On the other hand, Figure 6.6 shows the distribution of close scores. It can be seen that there are more edges when the close score is lower than 0.01 and higher than 0.04. Similarly, the distribution of close scores are less sensitive with P_{loss}.

Once we make sure that the distributions of similar and close scores, the following task is to verify the correctness of the selected thresholds. As mentioned above, the selection of thresholds will affect which trajectories will be grouped into a cluster. Since we are aware of the moving behaviors of trajectories, we try all possible values of λ and μ in a brute force manner and check whether the trajectories with the same moving behavior can be put into the same cluster or not. Table 6.2.2 shows the experimental results. The values of λ and μ which can induce each cluster containing the trajectories with the same moving behavior are usually in an interval. For example, when there are 80 trajectories, when λ is set between 0.01 and 0.02 and μ is set between 0.016 and 0.02, all clusters contain trajectories with the same moving behavior. The second and third columns show the ranges for λ and μ. The values of λ and μ decided by our heuristic approach is shown in the fourth and fifth columns. In each case, it can be seen the values of λ and μ obtained by our heuristic approach are fallen in the intervals which can induce the correct clustering results. Therefore, we can conclude that the proposed heuristic approach for threshold selection is effective.

6.2.3 Effectiveness of Algorithm ClusDG

Algorithm ClusDG is used to cluster trajectories with the same moving behaviors. The effectiveness of algorithm ClusDG leads the correctness of the derived frequent regions. Hence, in this experiment, we show the effectiveness of algorithm ClusDG under both datasets. Since the moving behaviors of trajectories are known, we can evaluate the percentage of trajectories in the right cluster. Specifically, the *successful rate* is defined as the number of trajectories in the correct cluster to the total number of trajectories. In both datasets, S refers the given trajectories and SP as the trajectories derived from the given ones with the loss rate varied.

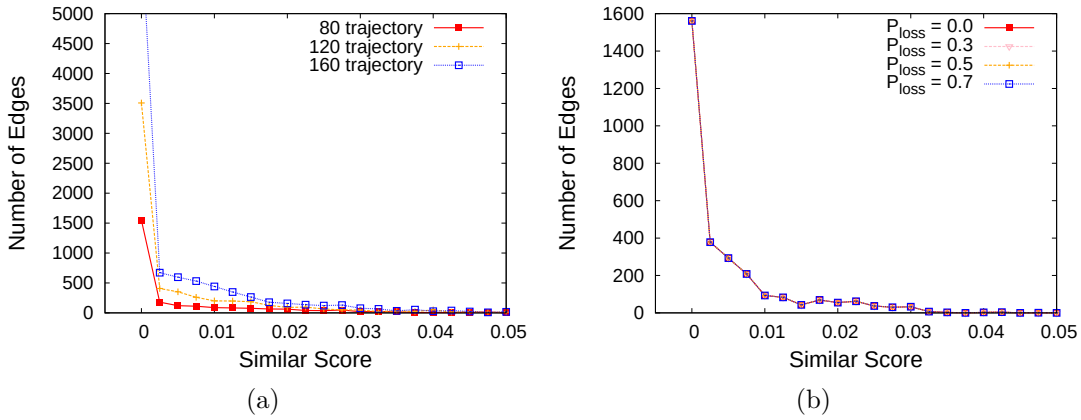


Figure 6.5: Similar scores on synthesis datasets.

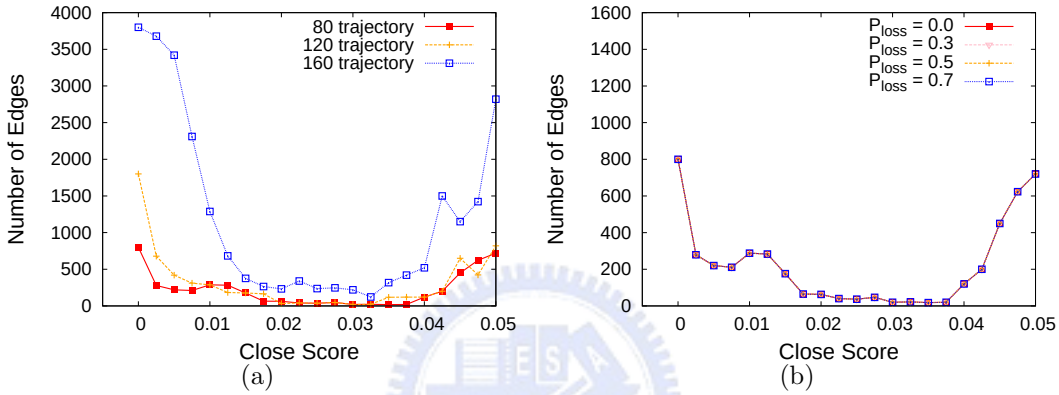
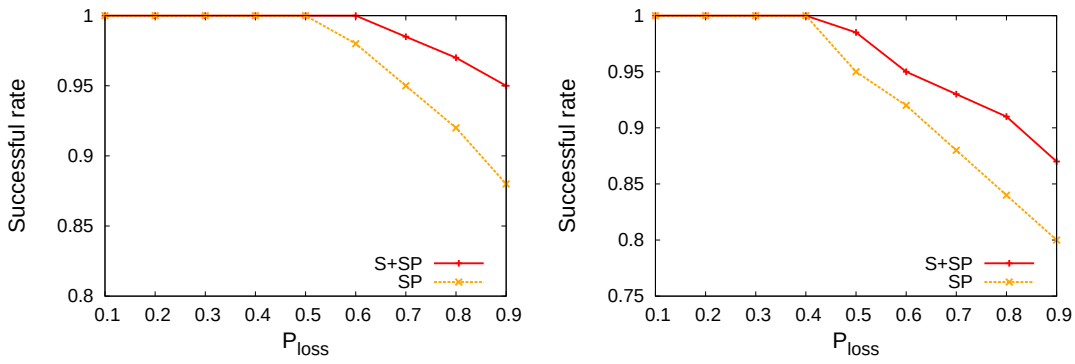


Figure 6.6: Close scores on synthesis datasets.

We compare the successful rate by the union SP and S (abbreviated as $S + SP$), and SP with different loss rate. The experimental results are shown in Figure 6.7, where the successful rate is above 80% in each set of trajectories. In both datasets, the successful rates of $S + SP$ is higher than that of SP and the decay speed of $S + SP$ is much slower than that of SP. The more complete trajectories the given set owns, the higher correctness of clustering the Algorithm ClusDG can achieve. It is because the more complete trajectories can not only form larger cores but also provide more hints for clustering those trajectories. From this example, we can conclude that trajectories with the same moving behavior can be grouped into a cluster effectively by algorithm ClusDG.

6.2.4 The Impact of Thresholds

In our approach, spatial-bias threshold ϵ is a user-specified threshold for defining a decaying function in the point-to-point similarity and bounding the size of a region. Also, temporal-bias threshold τ should be given in our approach for tolerating the time bias between points in the



(a) (b)

Figure 6.7: Successful rate with P_{loss} varied in (a) real and (b) synthesis datasets.

point-to-trajectory similarity. Here, we investigate the impact with two thresholds varied.

Figure 6.8 shows the precision and recall with ϵ varied. It can be seen that both precision and recall keep almost constant until $\epsilon = 1500$. With $\epsilon > 1500$, the precision and recall both decrease by 10%. It is because two trajectories have higher probability to identify as similar when ϵ becomes larger and larger. We observe that the clustering results are correct when $\epsilon < 1500$ (i.e., trajectories with the same moving behavior belong to the same cluster). However, when $\epsilon > 1500$, some distant trajectories with the different moving behaviors are put into a cluster such that the derived regions become less precise. It is interesting that the precision and the recall do not decrease when ϵ is small. One may claim that when the ϵ become smaller, two trajectories are hard to be similar or close to each other such that the trajectories with similar moving behaviors are also hard to group in a cluster. With fewer trajectories in a cluster, the information of a moving behavior may not be compensated. However, the value of λ and μ can be adjusted adaptively. Figure 6.9 shows the change of λ and μ values with ϵ varied. Note that if the value of ϵ is smaller than 5, it is hardly able to capture the moving behavior because r is set to be 5 meters. The values of λ and μ keep increasing with the value of ϵ smaller than 10. After that, the values of them remain constant. It shows that the dynamic adjustment of λ and μ can keep the precision and recall even if the value of ϵ is set as a unreasonable value.

On the other hand, the threshold of τ is used to tolerate the existence of a certain bias between trajectories. Figure 6.10 shows the precision and recall with τ varied. It can be seen that the precision and recall remain constant in all cases. Interestingly, although precision and recall are not sensitive with τ, various setting of τ affects the number of clusters. Note that the trajectories in synthesis datasets follow four kinds of moving behaviors. We observe that the number of clusters becomes 6 if $t \leq \tau$, where two clusters with two moving behaviors are

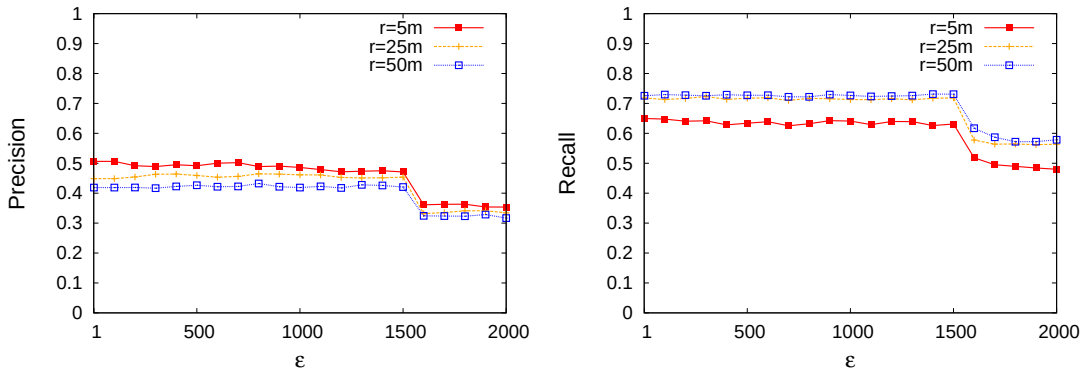


Figure 6.8: The impact of precision and recall in different ϵ.

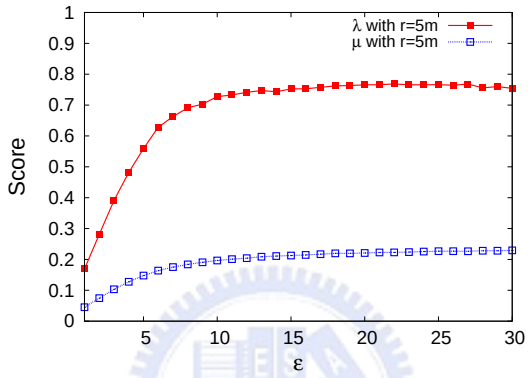


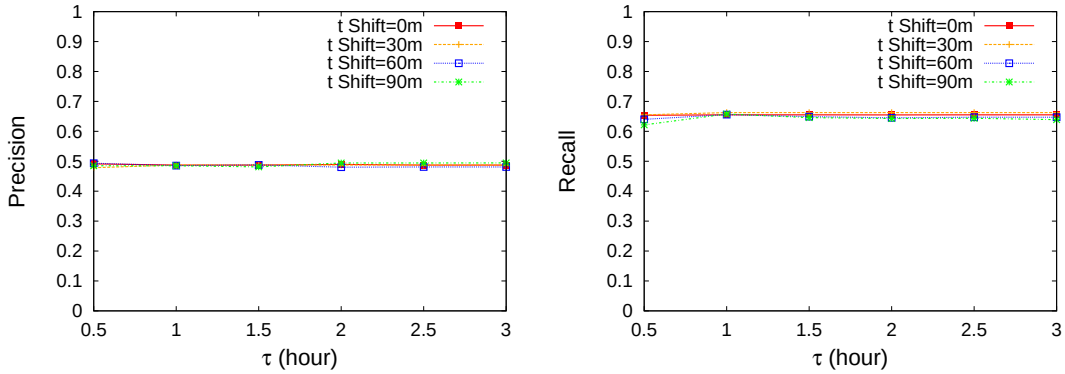
Figure 6.9: The values of λ and μ with ϵ varied.

divided into four smaller clusters with two moving behaviors. It will not affect the precision and recall because each two smaller clusters can be used to conduct the same moving behaviors.

Overall, besides the impacts of thresholds mentioned above, we can also observe that the values of precision and recall highly depends on the clustering results. The effectiveness of proposed clustering algorithm is shown from experimental results above. Therefore, our approach can perform well in our experiments.

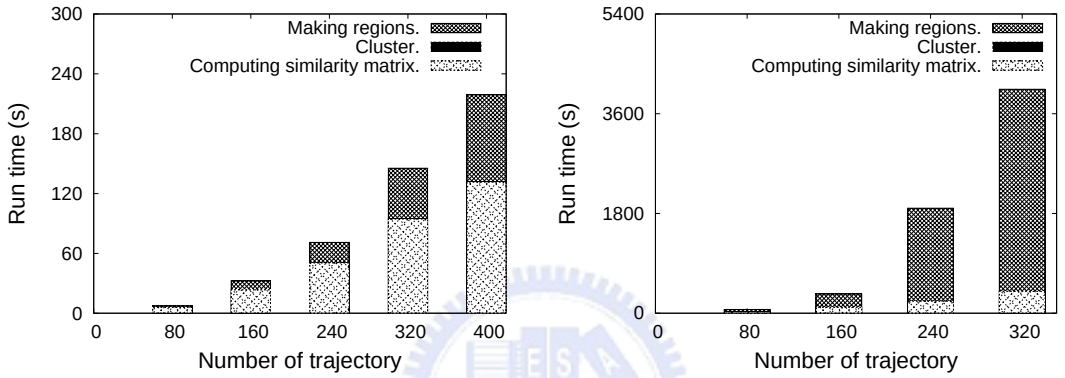
6.2.5 Execution Time

The execution time with the number of trajectories varied are discussed here. Two kinds of synthesis trajectories are used here where each trajectory has about 2000 points in short trajectories , and has 6000 points in long trajectories. The execution time includes three parts: 1. computing similarity between trajectories, 2. clustering trajectory, and 3. aggregating the information and generating regions. Figure 6.11 shows the experimental results. Overall, the execution time increases with the increasing of the number of trajectories. Moreover, the execution time for clustering is much less than others since the total number of trajectories



(a) (b)

Figure 6.10: The impact of precision and recall in different τ.



(a) (b)

Figure 6.11: Execution time under (a) short and (b) long trajectories.

is relatively fewer than the total number of points. However, the proportion of computing similarity and generating regions are not the same in different kinds of trajectories. In the short trajectories, as shown in Figure 6.11(a), the execution time for computing similarity is the major part in the whole execution time. In this case, the trajectories are shorter such that the aggregating information and making regions do not waste too much time. On the contrary, in Figure 6.11(b), our approach spends most execution in making regions when we consider the longer trajectories. Longer trajectories needs more time for aggregating the information and executing Douglas-Peucker line simplifier.

6.3 Conclusions

Nowadays, many positioning devices and techniques are more and more popular such that there are a lot of trajectories of people or vehicles can be easily obtained. From such a huge amount of trajectories collected, discovering trajectory patterns can benefit many potential and novel applications. In general, trajectory patterns indicate sequences of frequent regions

that a user usually appears. One of the challenge issues in trajectory pattern mining is how to define frequent region units in trajectory patterns. Previous works for determining frequent regions in trajectory patterns can be generally classified into two categories: the density-based approach and the line-based approach. However, the original trajectory data will have an impact on the determination of frequent regions. In reality, there are many factors, such as sampling method, sampling frequency and device constraints, will affect the capability of original trajectory data capturing the actual movements. Thus, if the original trajectory data only coarsely capture actual movements of a user, prior works mentioned above cannot accurately identify frequent regions, let alone deriving trajectory patterns. However, even if trajectories can only reflect partial movements of a user, they reveal some *clues* about the moving behaviors hidden in trajectories. Consequently, in this paper, given a set of trajectories, we propose an algorithm CACT (standing for Clustering and Aggregating Clues of Trajectories) for discovering trajectory patterns by exploiting such 'clues'. Exploiting the clues of trajectories, we first propose the similarity measurement for two trajectories by tolerating certain spatiotemporal bias. Furthermore, to deal with the existence of multiple moving behaviors in trajectories, we propose a clustering algorithm to divide trajectories into several groups. Trajectories in the same group reflect the same moving behavior of a user and the number of groups is the number of moving behaviors of a user. Then, for each group, we further propose an algorithm to derive a sequence of frequent regions with their corresponding representative line segments. To the best of our knowledge, this is the first work that claims to cluster trajectories into groups first and then derive the corresponding frequent regions within each group. Because of the design, our proposed method of mining trajectory patterns is able to handle the partial information of trajectories (i.e., the clues) and avoid the inaccuracy problem of frequent region determination. Through experimental studies on both synthetic and real datasets, we show that our approach is able to efficiently and effectively derive trajectory patterns to capture the frequent moving behaviors from trajectories.

Bibliography

[1] EveryTrail - GPS Travel Community. *[available] http://www.everytrail.com/.*

[2] MapMyRun Website. *[available] http://www.mapmyrun.com.*

[3] Run GPS Community Server. *[available] http://www.gps-sport.net/.*

[4] L. C. 0002 and R. T. Ng. On The Marriage of Lp-norms and Edit Distance. In *Proc. of VLDB*, 2004.

[5] A. Bhattacharya and S. K. Das. LeZi-Update: An Information-Theoretic Framework for Personal Mobility Tracking in PCS Networks. *Wireless Networks*, 8(2-3), 2002.

[6] H. Cao, N. Mamoulis, and D. W. Cheung. Mining Frequent Spatio-Temporal Sequential Patterns. In *Proc. of ICDM*, 2005.

[7] L. Chen, M. T. Özsu, and V. Oria. Robust and Fast Similarity Search for Moving Object Trajectories. In *Proc. of SIGMOD*, 2005.

[8] F. Giannotti, M. Nanni, and D. Pedreschi. Efficient Mining of Temporally Annotated Sequences. In *Proc. of SDM*, 2006.

[9] F. Giannotti, M. Nanni, F. Pinelli, and D. Pedreschi. Trajectory Pattern Mining. In *Proc. of KDD*, 2007.

[10] J. Gramm, J. Guo, F. Huffner, and R. Niedermeier. Data Reduction, Exact, and Heuristic Algorithms for Clique Cover. In *Proc. of SIAM Workshop on Algorithm Engineering and Experiments*, 2006.

[11] H. Jeung, Q. Liu, H. T. Shen, and X. Zhou. A Hybrid Prediction Model for Moving Objects. In *Proc. of ICDE*, 2008.

[12] H. Jeung, H. T. Shen, and X. Zhou. Mining Trajectory Patterns Using Hidden Markov Models. In *Proc. of DaWaK*, 2007.

[13] P. Kalnis, N. Mamoulis, and S. Bakiras. On Discovering Moving Clusters in Spatio-temporal Data. In *Proc. of SSTD*, 2005.

[14] E. J. Keogh. Exact Indexing of Dynamic Time Warping. In *Proc. of VLDB*, 2002.

[15] C.-H. Lo, W.-C. Peng, C.-W. Chen, T.-Y. Lin, and C.-S. Lin. CarWeb: A Traffic Data Collection Platform. In *Proc. of MDM*, 2008.

[16] S. Ma, S. Tang, D. Yang, T. Wang, and C. Yang. Incremental Maintenance of Discovered Mobile User Maximal Moving Sequential Patterns. In *Proc. of DASFAA*, 2004.

[17] N. Mamoulis, H. Cao, G. Kollios, M. Hadjieleftheriou, Y. Tao, and D. W. Cheung. Mining, Indexing, and Querying Historical Spatiotemporal Data. In *Proc. of KDD*, 2004.

[18] W.-C. Peng and M.-S. Chen. Developing Data Allocation Schemes by Incremental Mining of User Moving Patterns in a Mobile Computing System. *IEEE Transaction Knowledge and Data Engineering*, 15(1), 2003.

[19] W. J. Reed. The Pareto, Zipf and other power laws. *Economics Letters*, 74(1):15–19, December 2001.

[20] G. Trajcevski, H. Ding, P. Scheuermann, R. Tamassia, and D. Vaccaro. Dynamics-aware similarity of moving objects trajectories. In *Proc. of GIS*, 2007.

[21] F. Verhein. k-STARs: Sequences of Spatio-Temporal Association Rules. In *Proc. of ICDM Workshops*, 2006.

[22] F. Verhein and S. Chawla. Mining Spatio-temporal Association Rules, Sources, Sinks, Stationary Regions and Thoroughfares in Object Mobility Databases. In *Proc. of DASFAA*, 2006.

[23] H.-K. Wu, M.-H. Jin, and J.-T. Horng. Personal Paging Area Design Based On Mobiles Moving Behaviors. In *Proc. of INFOCOM*, 2001.

[24] B.-K. Yi, H. V. Jagadish, and C. Faloutsos. Efficient Retrieval of Similar Time Sequences Under Time Warping. In *ICDE*, 1998.