Mining Trajectory Patterns Using Hidden Markov Models

Hoyoung Jeung, Heng Tao Shen, and Xiaofang Zhou

National ICT Australia (NICTA), Brisbane, QLD, Australia
School of Information Technology and Electrical Engineering
The University of Queensland
{hoyoung,shenht,zxf}@itee.uq.edu.au

Abstract. Many studies of spatiotemporal pattern discovery partition data space into disjoint cells for effective processing. However, the discovery accuracy of the space-partitioning schemes highly depends on space granularity. Moreover, it cannot describe data statistics well when data spreads over not only one but many cells. In this study, we introduce a novel approach which takes advantages of the effectiveness of space-partitioning methods but overcomes those problems. Specifically, we uncover frequent regions where an object frequently visits from its trajectories. This process is unaffected by the space-partitioning problems. We then explain the relationships between the frequent regions and the partitioned cells using *trajectory pattern models* based on hidden Markov process. Under this approach, an object's movements are still described by the partitioned cells, however, its patterns are explained by the frequent regions which are more precise. Our experiments show the proposed method is more effective and accurate than existing space-partitioning methods.

1 Introduction

We are facing an unprecedented proliferation of mobile devices, many equipped with positional technologies such as GPS. These devices produce a huge amount of trajectory data which is described as geometry changes over time continuously. Since a large amount of trajectories can be accumulated for a short period of time, many applications need to summarize the data or extract valuable knowledge from it. As a part of the trend, discovery of trajectory patterns has been paid great attention due to many applications.

For the pattern discovery of spatiotemporal data, many techniques in the literature have partitioned data space into disjoint cells (e.g., fixed grid) [1,2,3,4]. The reasons are mainly two-folded. First, space-decomposition techniques bring efficiency of discovery process. Obviously, dealing with symbols identifying each cell is much simpler than handling real coordinates which should have bigger data size and give lower intuitions for data processing. Second, spatiotemporal data has a distinct characteristic from general data for mining studies (e.g., basket data). Assume Paul arrives at his work at 9 a.m. every weekday. Though his work is an identical place in semantic, the location may not be expressed by spatially the exact same coordinates because of the different vacancy of parking lots everyday. Therefore, pattern discovery methods need to regard slightly different locations as the same.

Despite the popularity of the *space-partitioning approaches*, it has two critical shortcomings. First, it cannot solve the *anwer loss* problem [5]. Suppose a problem finding dense regions in Figure 1. Data space is divided into nine cells from A to I and four objects (o_1, o_2, o_3, and, o_4) have moved in the space for two timestamps. If we define a dense region as a cell having more than two hitting points, r_1 cannot be a dense region though there are three close points since the points spread over three cells. Second, space-partitioning approaches have granularity problems. The precision of pattern discovery highly depends on how big or small the space divided. Especially, when there are many noises of movements (e.g., Paul unusually makes a trip to a far away for a few days), discovery process should manage a large size of data space, and thus the accuracy can decrease for efficient computation. For instance, r_2 and r_3 are distinct dense regions, however, both should be expressed by only one cell E due to the rough granularity.

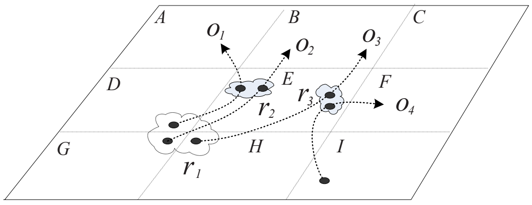

Fig. 1. Deficiencies of space-partitioning approaches

In order to overcome those problems, this study introduces a novel approach that takes both advantages of space-partitioning schemes and data-centric methods. Specifically, we reveal *frequent regions* that an object frequently visits by applying periodic data mining techniques [6] based on the data-centric approach. It is unaffected by space partitioning problems, hence, it should be more precise. However, it does not have the space-partitioning efficiency. For efficient data handling, we introduce *trajectory pattern model* (TPM) that explains the relationships between the regions and partitioned cells using hidden Markov models (HMMs). An HMM is a doubly embedded stochastic process with an underlying stochastic process that is not observable. We model partitioned cells to observable states and discovered frequent regions to hidden states. Therefore, the TPM let applications be able to deal with symbols of the cells (instead of using real coordinates) for effectiveness but have more precise discovery results than existing space-partition methods.

Building a TPM from historical trajectories can be useful for many applications. First, it computes the probability of a given observation sequence. It implies the TPM can explain how the current movements (a sequence of cell symbols) of an object is similar to its common movement patterns (a sequence of frequent regions). Second, given a cell symbol sequence, it can also compute the most likely sequence of frequent regions. Moreover, the model can be trained by newly added data. Hence, it can reflect not only historical movements of an object but also its current motion trends.

2 Related Work

In this Section, we study previous work based on two major discovery techniques, *Markov chain models* and *spatiotemporal data mining*, for extracting movement patterns of an object from historical trajectories.

Markov chain models have been widely used in order to estimate the probability of an object's movements from one region or state to another at next time period. Ishikawa et al. derive the Markov transition probabilities between cells from indexed trajectories [1]. In their further study [7], a special type of histogram, called mobility histogram, is used to describe mobility statistics based on the Markov chain model. They also represent the histogram as cube-like logical structures and support an OLAP-style analysis. Authors in [8] classify an object's mobility patterns into three states (stationary state, linear movement, and random movement) and apply Markov transition probabilities to explain a movement change one state to another. [9,10] consider the location tracking problem in PCS networks. Both studies are based on the same Markov process in order to describe users' movements from one or multiple PCS cells to another cell. However, they have different ways to model users' mobilities using Morkov models, thus, show distinct results to each other.

Spatiotemporal data mining methods have been also studied well for describing objects' patterns. [2] introduces mining algorithms that detect a user's moving patterns, and exploits the mined information to invent a data allocation method in a mobile computing environment. Another mining technique is shown in [11]. This study focuses on discovering spatio-temporal patterns in environmental data. In [6], authors do not only explore periodic patterns of objects but also present indexing and querying techniques of the discovered or non-discovered pattern information. [3,4] address *spatio-temporal association rules* of the form $(r_i, t_1, p) \longrightarrow (r_j, t_2)$ with an appearance probability p, where r_i and r_j are regions at time (interval) t_1 and t_2 respectively $(t_2 > t_1)$. It implies that an object in r_i at time t_1 is likely to appear in r_j at time t_2 with $p\%$ probability. Besides, [3] considers spatial semantic areas (i.e. sources, sinks, stationary regions, and thoroughfares) in each r_i as well as more comprehensive definitions and algorithms of spatio-temporal association rules.

All above studies except [6] are based on the space-partitioning schemes, thus, the discovery accuracy depends on how the system decides space granularity of data space. When they partition the data space into a large number of small size cells, the accuracy increases, however, managing such many cells in memory can be burden to the system. On the contrary, using large size cells for partitioning cause low precision of discovery. Moreover, they cannot avoid the answer-loss problem no matter how the spatial granularity is set to.

3 Problem Definition

We assume a database stores a large volume of accumulated trajectories and each location in each trajectory is sampled periodically with a discrete time interval. Let a point $p_i = (l_i, t_i)$ represent a d-dimensional location l_i at time t_i. A whole trajectory T of an object is denoted as $\{p_0, p_1, \cdots, p_{n-1}\}$, where n is the total number of points. The entire data space is partitioned into k fixed grid cells $C = c_1, c_2, \cdots, c_k$, each cell c_i of which has the same coverage size.

The goals of our study are two-folded. First, we aim to reveal spatial regions, namely *frequent regions*, where an object frequently and periodically appears. For example, given a trajectory having 10 days movements, we are interested in discovering areas where the object appears more than 5 days at the same time. Specifically, given an integer P, called *period*, we consider decomposing T into $\lfloor \frac{n}{P} \rfloor$ sub-trajectories and group points G_w having the same time offset w of P $(0 \leq w < P)$ in each sub-trajectory. We formally define the frequent region as follows:

Definition 1. *A **frequent region** is a minimum bounding rectangle that consists of a set of points in G, each point of which contains at least MinPts number of neighborhood points in a radius Eps.*

Second, building hidden Markov models to explain relationships between frequent regions and partitioned cells is another goal in this study. A discrete Markov process is a stochastic process based on the Markov assumption, under which the probability of a certain observation only depends on the observation that directly preceded it. It has a finite number of states that make transitions according to a set of probabilities associated with each state at discrete times. A hidden Markov process is a generalization of a discrete Markov process where a given state may have several existing transitions, all corresponding to the same observation symbol [12]. In other words, the stochastic process is hidden and can only be observed through another set of stochastic processes that produce the sequence of observations.

There are many possible ways to model an object's mobility using hidden Markov models. Our approach is to construct a trajectory pattern model where each state corresponds to a frequent region. Observations are cells that change the current state with some probability to a new state (a new current frequent region). Formally,

Definition 2. *A **trajectory pattern model** describes an object's movement patterns based on hidden Markov process. It consists of a set of N frequent regions, each of which is associated with a set of M possible partitioned cells.*

4 Discovery of Trajectory Pattern Models

In this Section, we describe how to solve the two sub-problems of Section 3, which are discovery of frequent regions and building trajectory pattern models.

4.1 Extracting Frequent Regions

For the detection of frequent regions from an object's trajectories, we adopt a periodical pattern mining methods [6]. In these techniques, the whole trajectory is decomposed into $\lfloor \frac{n}{P} \rfloor$ sub-trajectories. Next, all locations from $\lfloor \frac{n}{P} \rfloor$ sub-trajectories which have the same time offset w of P will be grouped into one group G_w. P is data-dependent and has no definite value. For example, $P =$ 'a day' in a traffic control application since many vehicles have daily patterns, while animal's annual migration behaviors can be discovered by $P =$ 'a year'. A The clustering method DBSCAN [13] is then applied to find the dense clusters R_w in each G_w. Given two parameters *Eps* and *MinPts*, DBSCAN finds dense clusters, each point of which has to contain at least *MinPts* number of neighborhood points within a radius *Eps*.

In spite of many advantages of DBSCAN, it can not be applicable to our study directly. Our key methods of this research is to describe correlations between frequent regions, and between partitioned cells and frequent regions. It implies the size of a frequent region can affect explanation of the patterns considerably. Therefore, we decompose a cluster when it is 'large'. As each dataset may have different data distributions and map extents from others, a proper area size for cluster decompositions can not be easily detected. Due to this property, we use $(Eps * \kappa)^2$ to define a 'large' cluster, where κ is given by a user (named *area limiter*). κ limits a cluster's size merely with respect to *Eps*. For example, $\kappa = 2$ means that a cluster's size must be smaller than the square's area having double *Eps* length on a side. Thus, a user does not need to know about the data distributions. In our experiments, when $5 \leq \kappa \leq 10$ regardless of data characteristics, the cluster decomposition showed more precise discovery results.

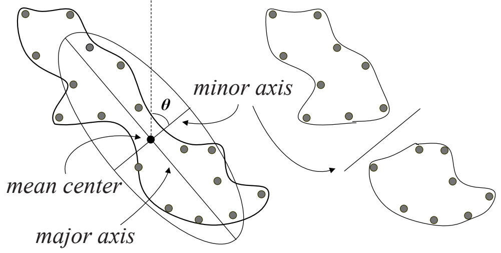

Fig. 2. Cluster decomposition using standard deviation ellipse

When partitioning is needed on a cluster we apply *standard deviation ellipse* for the decomposition process. The standard deviation ellipse captures the directional bias in a point distribution using three components: major (longer) axis, minor (shorter) axis, and θ between the north and the y axis rotated clockwise. Figure 2 illustrates how a cluster is partitioned into two co-clusters based on the standard deviation ellipse. This method first computes the minor axis of the cluster's standard deviation ellipse. Next, it classifies each point dividing the cluster into two co-clusters. This classification is performed by checking whether each point is left (above) of the minor axis or right (bellow). The shapes of co-clusters decomposed tend to be sphere-like, hence, the center position of each co-cluster can represent its cluster effectively. The partitioning method also distributes an almost equal number of points to each co-cluster. When the number of points in any co-cluster is smaller than $MinPts$, we cease the cluster partitioning. The reason for this is that a cluster having less than $MinPts$ points contradicts the definition of a frequent region and becomes less useful. The whole process of cluster decomposition is described in the Algorithm 1:

Algorithm 1. Cluster Decomposition

```
Input:
    given cluster $C$, area limiter $\kappa$, radius $Eps$, minimum number of points $MinPts$
Output:
    a set of clusters
Description:
 1: if area of $C \geq (Eps \times \kappa)^2$ then
 2:    get the minor axis $minx$ of $C$'s standard deviation ellipse
 3:    for each point $p$ in $C$ do
 4:       if $p$ is the left of $minx$ then
 5:          add $p$ to a co-cluster $c1$
 6:       else
 7:          add $p$ to a co-cluster $c2$
 8:    if $numPoints\ (c1) \geq MinPts$ and $numPoints\ (c2) \geq MinPts$ then
 9:       return $c1$ and $c2$
10: return $C$
```

4.2 Building Trajectory Pattern Models

Let us recall the movements of objects in Figure 1. Assume an application needs to find an object which has the most similar movements to o_2. Though o_1 is obviously the closest answer, it is hard to be detected since there is only one common cell between $o_2 = GEB$ and $o_1 = DEA$, which has the same number of common cell as between o_2 and o_4. In our approach, *trajectory pattern model*, we illustrate real movements of an object as a sequence of frequent regions instead of a cell sequence. For example, $o_1 = r_1r_2*$, $o_2 = r_1r_2*$, $o_3 = r_1r_3*$, and $o_4 = *r_3*$ ('$*$' for non-frequent regions). We then apply hidden Markov models to figure out relationships between a cell sequence (e.g., ABC) and a frequent region sequence (e.g., r_1r_2*). Under this approach, although an object's movements are denoted as a cell sequence, the problem of finding most similar movements to o_2 can be performed on the frequent region sequences, which are more precise.

In order to construct a trajectory pattern model, we must define the elements of an HMM, which are (i) the number of states N, (ii) the number of observation symbols per state M, (iii) an initial state π, (iv) a set of state transition probabilities A, and (v) a set of observation symbol probabilities B. This HMM parameter set can be denoted as $\lambda = (A, B, \pi)$. How to model this λ is critical to explain an object's mobilities since there are many possible ways to model.

Our approach is to construct λ where each state corresponds to a frequent region r_i when a cell c_i appears. Therefore, hidden states (frequent regions) are expressed by

$$S = \{r_{i_1}, r_{i_2}\ r_{i_3}, \cdots, r_{i_N}\},$$

and the number of states N is the number of r_i when c_i is found. We also define an observation sequence O as

$$O = \{c_{i_1}, c_{i_2}\ c_{i_3}, \cdots, c_{i_M}\}$$

where M corresponds to the total number of partitioned cells. The state transition probabilities can be formed as a matrix $A = \{a_{ij}\}$ where

$$a_{ij} = P[r_{t+1} = S_j | r_t = S_i], 1 \leq i, j \leq N,$$

and r_t denotes the actual state at time t in the sequence S. In the same manner, the observation transition probabilities B is expressed by following matrix

$$b_{ij} = P[c_t | r_t = S_i], 1 \leq i \leq N, 1 \leq j \leq M$$

with elements b_{ij} denoting the probability of observing symbol $j \in [1, M]$ that the system is in state $i \in [1, N]$. The initial state is defined as $\pi = \{\pi_i\}$ $(1 \leq i \leq N)$ which π_i describes the distribution over the initial frequent region set.

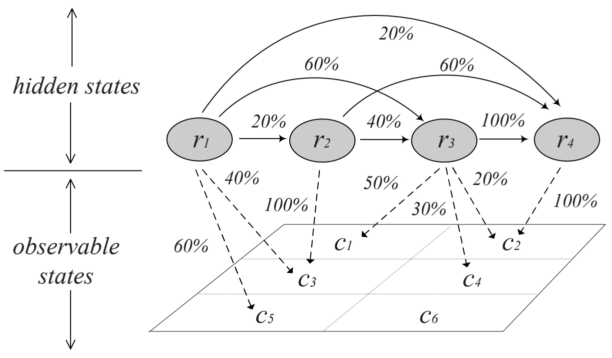

Fig. 3. An example of a trajectory pattern model

Figure 3 represents an example of a TPM modelling based on our scheme. Circles stand for frequent regions as hidden states and the grid does partitioned cells as observation symbols. In the example, the TPM parameters can be denoted as follows; hidden states $S = \{r_1, r_2, r_3, r_4\}$, thus the number of states $N = 4$, and observable symbols $O = \{c_1, c_2, c_3, c_4, c_5, c_6\}$ $(M = 6)$. Transition probabilities of states and observation symbol probabilities are illustrated as follows:

$$A = \{a_{ij}\} = \begin{bmatrix} 0 & 20 & 60 & 20 \\ 0 & 0 & 40 & 60 \\ 0 & 0 & 0 & 100 \\ 0 & 0 & 0 & 0 \end{bmatrix}, \qquad B = \{b_{ij}\} = \begin{bmatrix} 0 & 0 & 40 & 0 & 60 & 0 \\ 0 & 0 & 100 & 0 & 0 & 0 \\ 50 & 20 & 0 & 30 & 0 & 0 \\ 0 & 100 & 0 & 0 & 0 & 0 \end{bmatrix}$$

In order to compute A, we investigate whether each point p_w of a sub-trajectory is geometrically within any of r_i for all p_i. If so, we report the frequent region id otherwise a wild card mark '*'. Therefore, a sub-trajectory can be denoted by a state sequence as $r_1 r_2 * r_4$. We then compute the probabilities of each state transition from the sequences. Likewise, elements of π are calculated by the same way. The computation of B is done by checking if p_i belongs to both r_i and c_i or c_i only.

An excellent advantage of TPMs is that λ can adjust its parameters (A, B, π) to maximize $P(O|\lambda)$ using an iterative procedure. It implies that the model can be trained more powerfully by compiled trajectories of an object as time goes on. Hence, it can reflect the trends of the object's motions precisely.

5 Experiments

The experiments in this study were designed for comparison of discovery accuracy and effectiveness of data management between a space-partitioning method and our scheme. In order to evaluate them, we implemented a popular method of spatiotemporal data mining using observable Markov models (OMM), which was based on the space-partitioning scheme. We also did our approach, trajectory pattern models (TPM), using hidden Markov models. Both methods were implemented in the C++ language on a Windows XP operating system and run on a dual processor Intel Pentium4 3.0 Ghz system with a memory of 512MB.

Due to the lack of accumulated real datasets, we generated three synthetic datasets having different number of sub-trajectories and timestamps (Bike-small had 50 sub-trajectories and 50 timestamps, 200 and 200 for Bike-medium, and 500 and 500 for Bike-large). For the generation, we developed a data generator that produced similar trajectories to a given trajectory. We used a real trajectory as an input of the generator, which was measured by a GPS mounted bicycle over an eight hour period in a small town of Australia. The extent of each dataset was normalized to [0,10000] in both x and y axis.

In the first set of experiments, we studied how the space granularity affected discovery accuracies of OMM and TPM. To measure them, we computed the distances between each actual point of the input trajectory for the dataset generation and the center point of a cell for OMM, which the actual point belonged to. We did the same process for obtaining TPM's errors.

Figure 4 demonstrates the changes of discovery accuracies along the number of partitioned cells. As expected, the errors of OMM were significant when the number of cells were small since one cell had a too wide coverage to describe

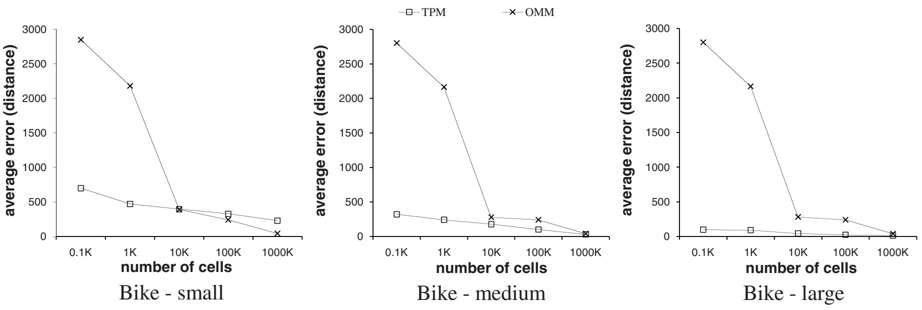

Fig. 4. Discovery accuracy changes along the number of partitioned cells

details of the object's movements. On the contrary, our approach was slightly affected by the space granularity. An interesting observation was that TPM's errors decreased as the dataset size grew. The reason was because TPM had more trajectories in bigger datasets that can obtain stronger and more precise trajectory patterns.

Another aspect about space-partitioning techniques to be considered is that a large number of cells needs more complicated management schemes so as to handle the partition effectively. For example, the data distribution is skew, many cells are not related to objects' movements but they are still needed to be managed. Therefore, though a large number of cells has lower errors, it involves a bigger number of partition cells, which causes overheads of systems.

We also explored size changes of discovered trajectory patterns along different size of datasets. As shown in Figure 5, our method showed relatively small pattern size compared to OMM. In fact, TPM's size was computed by the number of transition items and probabilities for each matrix of (A, B, π), however, the system did not need to store all the transition probability values. Recall the example of TPM in Section 4.2. Due to the time order of frequent regions, the matrix of transition probabilities are always filled with zero values for more than half of the matrix (left-bottom part), which are not physically stored in the system.

On the contrary, discovered pattern sizes of OMM were very large especially when it had higher orders (i.e., $A \rightarrow B$: 1st order, $ABC \rightarrow D$: 3rd order, and $ABCDE \rightarrow F$: 5th order). Some applications need to handle higher order transitions, thus, the size of higher order OMM is also important. Our approach can handle various length of sequences (i.e., higher order chains) without any extra storage consumption. This means our method can be applicable to many applications that have storage limits.

6 Conclusion

In this paper, we addressed space-partitioning problems which were the spatial granularity problem and the answer loss problem. In fact, many spatiotemporal

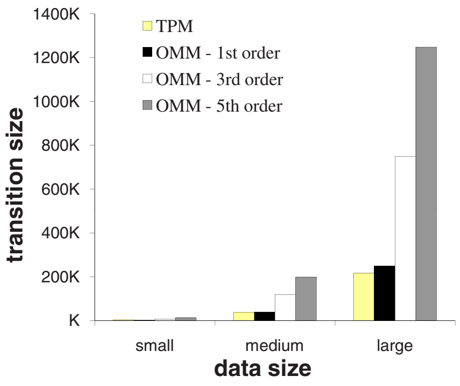

Fig. 5. Changes of trajectory pattern size along dataset size

mining methods are based on the space-partitioning scheme, thus, their pattern discoveries were not effective. In order to overcome the problems, we first revealed frequent regions where an object frequently visited by applying a data-centric approach. We then described the relationships between the partitioned cells and the frequent regions by using the *trajectory pattern models* based on hidden Markov process. Therefore, applications based on our scheme can still use space-partitions but have more precise discovery results. As shown in the experiments, our approach outperformed previous studies, especially when data space had a large size.

Acknowledgements. National ICT Australia is funded by the Australian Government's Backing Australia's Ability initiative, in part through the Australian Research Council.

References

1. Ishikawa, Y., Tsukamoto, Y., Kitagawa, H.: Extracting mobility statistics from indexed spatio-temporal datasets. In: STDBM, pp. 9–16 (2004)
2. Peng, W.C., Chen, M.S.: Developing data allocation schemes by incremental mining of user moving patterns in a mobile computing system. TKDE 15(1), 70–85 (2003)
3. Verhein, F., Chawla, S.: Mining spatio-temporal association rules, sources, sinks, stationary regions and thoroughfares in object mobility databases. In: DASFAA (2006)
4. Tao, Y., Kollios, G., Considine, J., Li, F., Papadias, D.: Spatio-temporal aggregation using sketches. In: ICDE, p. 214 (2004)
5. Jensen, C.S., Lin, D., Ooi, B.C., Zhang, R.: Effective density queries on continuously moving objects. In: ICDE, p. 71 (2006)
6. Mamoulis, N., Cao, H., Kollios, G., Hadjieleftheriou, M., Tao, Y., Cheung, D.W.: Mining, indexing, and querying historical spatiotemporal data. In: SIGKDD, pp. 236–245 (2004)
7. Ishikawa, Y., Machida, Y., Kitagawa, H.: A dynamic mobility histogram construction method based on markov chains. In: SSDBM, pp. 359–368 (2006)

8. Song, M., Ryu, J., Lee, S., Hwang, C.S.: Considering mobility patterns in moving objects database. In: ICPP, p. 597 (2003)
9. Yang, M.H., Chen, L.W., Sheu, J.P., Tseng, Y.C.: A traveling salesman mobility model and its location tracking in pcs networks. In: ICDCS, pp. 517–523 (2001)
10. Bhattacharya, A., Das, S.K.: Lezi-update: an information-theoretic approach to track mobile users in pcs networks. In: MobiCom, pp. 1–12 (1999)
11. Tsoukatos, I., Gunopulos, D.: Efficient mining of spatiotemporal patterns. In: SSTD, pp. 425–442 (2001)
12. Rabiner, L.: A tutorial on hidden markov models and selected applications in speech recognition. Proceedings of the IEEE 77, 257–286 (1989)
13. Ester, M., Kriegel, H.P., Sander, J., Xu, X.: A density-based algorithm for discovering clusters in large spatial databases with noise. In: SIGKDD, pp. 226–231 (1996)