Contents

11 Data Mining for Moving Object Databases **1**
Yoshiharu Ishikawa

11.1 Introduction 1
11.2 Mobility Prediction Using Movement Histories 2
11.3 Sequential Pattern Mining-based Approaches 6
11.4 Finding Other Interesting Patterns 8
11.5 Clustering Moving Objects 11
11.6 Dense Regions and Selectivity Estimation 14
11.7 Comparing Moving Object Trajectories 19
11.8 Conclusions 21
References 22

Index **28**

11 Data Mining for Moving Object Databases

Yoshiharu Ishikawa

Information Technology Center, Nagoya University
Furo-cho, Chikusa-ku, Nagoya, Japan

11.1 INTRODUCTION

Due to the recent developments in mobile devices and GPS systems and the progress of network technology, mobile computing has become a key technology today. Along with the progress in mobile technologies, research on moving object databases is currently being actively investigated in the area of database research [16]. As its name suggests, a *moving object database* is a database that stores and manages information on moving objects such as vehicles, pedestrians with mobile devices, and so on. Various research and development efforts regarding moving object databases have been conducted, including several topics such as data models for representing movement behaviors appropriately, query processing and indexing methods for answering queries efficiently, and application technologies that utilize the underlying moving object databases effectively.

In this chapter, we focus on *data mining* technology for moving object databases. Since the middle of the 1990s, data mining research has been growing rapidly and it has become one of the main research fields in computer science [25, 56]. Although data mining research has been expanding in a variety of fields in recent years, data mining technology for moving object databases is still in an emerging stage of development. However, it is highly promising because many moving objects can be monitored in real-time using current mobile information technologies. Since data managed in a moving object database is highly dynamic and have spatio-temporal semantics, new data mining technologies should be developed. This chapter presents a brief introduction to current trends in data mining on moving object databases including the author's own efforts in this area. We do not intend a complete survey and omit some topics of moving object databases such as data modeling, query processing, and indexing methods except for the issues related to data mining.

The organization of this article is as follows. Section 11.2 introduces some approaches to mobility prediction that can be considered as special cases of data mining for moving objects. Section 11.3 provides a brief survey of methods applying sequential pattern mining to moving databases. Section 11.4 presents other movement pattern mining techniques. Section 11.5 introduces techniques for clustering moving objects and Section 11.6 provides a short summary of density estimation and query selectivity estimation for moving object databases. Section 11.7 presents some interesting ideas for comparing trajectories. Finally, Section 11.8 concludes the article.

11.2 MOBILITY PREDICTION USING MOVEMENT HISTORIES

Mobility prediction, which is used for predicting the future trajectory of a given moving object, is a widely researched topic in mobile computing. Consider a mobile user in a cell-based mobile phone network who is moving continuously while making a call. The underlying network system has to transfer his calling status between cells [70]. If the next cell to which a mobile user will move can be predicted, then an efficient resource reservation and quick handover between base stations can be achieved. So far, various mobility prediction methods have been proposed. A study [6] provides a good survey of this topic. It roughly classifies the approaches to mobility prediction into two categories:

- *Domain-independent methods*: Locations or cells are treated as *symbols*, and only location names are considered, without taking other semantics into account.

- *Domain-specific methods*: Additional information, such as coordinates, directions, and velocities of moving objects, road networks and map information, and/or facility locations are used.

In this subsection, we introduce some selected mobility prediction models that utilize *movement histories*, since they are related to the concept of data mining. We focus in particular on Markov predictors and their extensions.

11.2.1 Domain-Independent Markov Predictors

We first describe the most basic type of mobility predictors, called Markov predictors, and their variants.

11.2.1.1 Markov Predictors

The underlying idea of *Markov predictors* is simple: the next location is predicted from recent movement history based on the notion of *Markov chains*. In this framework, each location is considered as a *state* and each movement between locations

corresponds to a *transition*. In an order-k Markov predictor, the k most recent locations in a movement history are used for prediction.

Suppose $X = (X_1, \ldots, X_n)$ is a sequence of random variables taking values in a finite set of locations $L = \{l_1, \ldots, l_m\}$. L represents the state space. The *Markov properties* are as follows:

$$\begin{aligned} \Pr(X_{n+1} = l_i | X_1, \ldots, X_n) &= \Pr(X_{n+1} = l_i | X_{n-k+1}, \ldots, X_n) \quad (11.1) \\ &= \Pr(X_{n+1} = l_i | X_{j+1}, \ldots, X_{j+k}) \quad (11.2) \end{aligned}$$

Equation (11.1) implies that the probability only depends on the most recent k movements and Eq. (11.2) indicates that the probability is *stationary*, or *time invariant*. The probability is basically estimated according to movement histories. The next location predicted is the location that maximizes the probability.

11.2.1.2 Applying String Compression Techniques

There is an approach to extending Markov predictors using the *string compression* technique to summarize statistics in a compact manner. The underlying idea is that a string compression method has the predictive ability to estimate which characters will follow when an input text is given.

In the following, we briefly illustrate a representative method called the *LZ-based predictor* that is an extended version of the order-k Markov predictor, although k is a variable that changes depending on the input. The method is based on the well-known *Lempel-Zip text compression* method (*LZ78*) [45]. LZ78 reads a text stream sequentially and constructs a *dictionary* with *trie* (or tree) structure to summarize the occurring patterns in an online manner. The dictionary is referenced while the compression is in progress. The idea of LZ78 predictors for mobility prediction is to consider a sequence of location symbols instead of a text stream and to use the constructed dictionary for the purposes of prediction.

For example, suppose the movement history of a moving object is given as "ABCABACBADABCD," where A to D are location symbols. For this input, LZ78 constructs a trie as shown in Fig. 11.1, where ε represents an empty character. The trie is constructed by partitioning the input into substrings such as "A/B/C/AB/AC/BA/D/ABC" while inserting them. The number shown in the upper-right of each node represents the node identifier and is referenced when the same substring occurs while reading the succeeding text. LZ78 encodes the example text as "0A0B0C1B1C2A0D4C." LZ-based predictors store additional information: each node contains a *counter* to record the number of visits. The count information is used for mobility prediction. For instance, if we want to estimate the probabilities that an object in A next moves to B or C, they are basically given as $\Pr(\mathtt{B}|\mathtt{A}) = 2/4$ and $\Pr(\mathtt{C}|\mathtt{A}) = 1/4$. B is, therefore, predicted as the next location.

The problem of LZ-based predictors is that information for some substrings on the boundaries is lost. For example, a substring "CBA" is contained in the input but does not appear in the trie. To cope with this problem, Bhattacharya and Das [3] proposed the *LeZi-Update* method. When inserting a substring into a trie, LeZi-Update also

inserts its suffixes. For example, when we insert "ABC" into the trie for the example text "A/B/C/AB/AC/BA/D/ABC," its proper suffixes "BC" and "C" are also inserted. The method then tries to cope with the cross boundary problem, but note that it does not solve all the boundary problems.

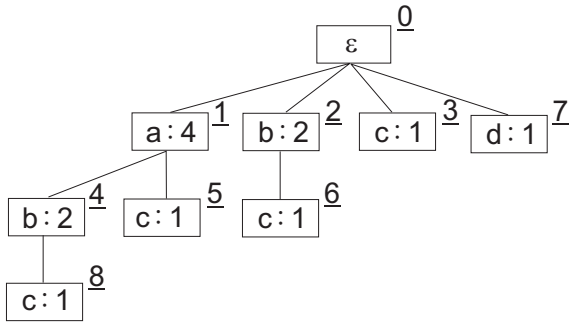

Fig. 11.1 Trie-structured dictionary of LZ-based predictors

Song et al. [54] collected a large user mobility dataset from an actual mobile wireless LAN and made an experimental evaluation of domain-independent predictors. Several methods including the Markov-based approach, the LZ-based approach and its variations, and two other related methods were compared. The results are quite interesting: the experiments show that the low-order Markov predictors perform as well or better than the more complex predictors. The best result is obtained by an order-2 Markov predictor with some enhancement. Although the LZ-based approach and other related approaches are effective for text data, the experimental results indicate that they are not necessarily effective for movement prediction. The reason is that the statistical properties of text data and movement histories are quite different.

11.2.2 Markov Chain Model Over Spatial Grid Cells

11.2.2.1 Basic Idea

Markov predictors are basically domain-independent, that is, they treat cells as symbols and do not use other information such as locations. The mobility model proposed by our group [22] extends the basic Markov chain model by incorporating spatial information directly. The fundamental difference is that we consider a spatial grid structure over the target space. Fig. 11.2 illustrates this concept. Each dimension of the target space is equally divided into 2^P ranges. The figure shows the case of $P = 2$. We call such partitioning *level-P partitioning*. Based on this partitioning method, there exist $R = 2^{2P}$ grid cells. For each region, a $2P$ bit grid cell number that satisfies the *Z-ordering method* [49] is assigned. The Z-ordering method has the advantage that close cells tend to have similar values. The figure shows that object A located in region 9 at $t = \tau$ moves to region 12 at $t = \tau + 1$ then moves to region 6

at $t = \tau + 2$. We denote the transition by $9^{(2)} \rightarrow 12^{(2)} \rightarrow 6^{(2)}$, where $^{(2)}$ means that the partition level is $P = 2$.

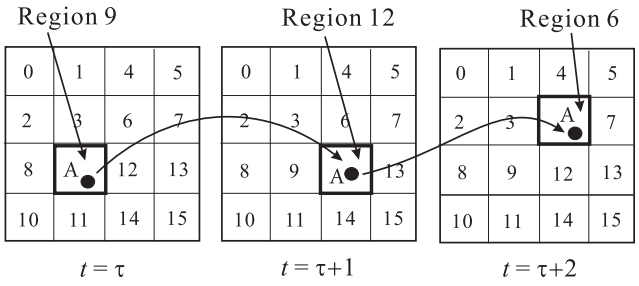

Fig. 11.2 Markov chain model over spatial grid cells

Suppose that another moving object `B` located in grid cell 9 moves to cell 12 after a unit of time. Supposing that we want to know the probability that object `B` moves to region 6 next, and that we denote the probability by $\Pr(6|9, 12)$. If we assume the transition between spatial grid cells satisfies the Markov property, we can say that the probability is a second-order Markov transition probability. We can generalize the idea to order-k Markov transition probability $\Pr(c_k|c_0, \ldots, c_{k-1})$, where c_i ($i = 0, \ldots, k$) are cell numbers.

11.2.2.2 Multiple Resolutions

An interesting feature of the model is that it allows multiple resolutions. When analyzing movement data, it is often necessary to view the data at different degrees of coarseness. For example, we may wish to analyze the overall trends at a coarse resolution and then focus on some specific regions and make detailed analyses at a fine resolution. The situation is similar to the case when we use the *drill-down* operation in *On-Line Analytical Processing* (*OLAP*) [25, 48]. The opposite operation from fine to coarse resolution corresponds to the *roll-up* operation.

We illustrate how to represent roll-up and drill-down operations in a spatial sense using Figure 11.3, where first-order Markov chains are assumed. The figure on the left-hand side shows the level-1 partitioning and the figure on the right-hand side shows the level-2 partitioning. The level-1 (level-2) partitioning is the "roll-up" ("drill-down") version of the level-2 (level-1) representation. It is easy to map representations at different resolutions because of the property of the Z-ordering method. For example, consider cell number 9 on the level-2 partitioning. Its binary representation is "`1001`." If we omit the last two digits, we get "`10`," which is the corresponding cell number 2 in the level-1 partitioning.

On the basis of the above idea, we have demonstrated a mobility histogram construction method that approximates the movement statistics in Ref. [22]. Its details are described in Subsection 11.6.3. We have also proposed an efficient algorithm to process queries on mobility statistics based on Markov chains [21].

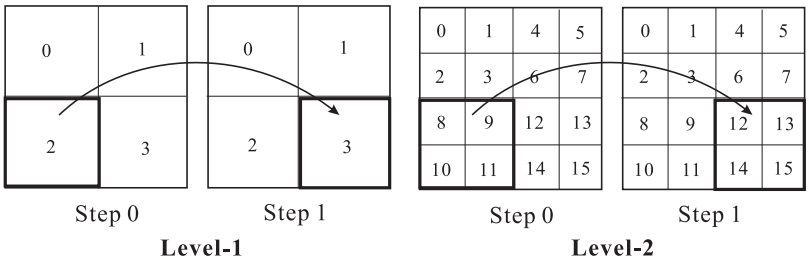

Fig. 11.3 Roll-up and drill-down

Its characteristic feature is the effective use of a spatial index to accelerate query processing.

11.3 SEQUENTIAL PATTERN MINING-BASED APPROACHES

Sequential pattern mining [2, 25, 56] is an important topic in data mining and is often applied to basket data analysis and Web usage analysis. There are some interesting applications of sequential data mining to moving object databases. First, we briefly explain the notion of sequential pattern mining.

Suppose that there are four moving objects and their movement histories are given as $h_1 = \langle \mathtt{A}, \mathtt{B}, \mathtt{C} \rangle$, $h_2 = \langle \mathtt{B}, \mathtt{C}, \mathtt{E} \rangle$, $h_3 = \langle \mathtt{A}, \mathtt{C} \rangle$, and $h_4 = \langle \mathtt{A}, \mathtt{C}, \mathtt{D}, \mathtt{C}, \mathtt{E} \rangle$, where A to E are location symbols and $\langle\ \rangle$ represents a sequence. For example, $\langle \mathtt{A}, \mathtt{B}, \mathtt{C} \rangle$ means the moving object visits locations A, B, and C in this order. An example of a *sequential pattern* is $\langle \mathtt{A}, \mathtt{C} \rangle$ meaning C is visited after A. Note that other places between A and C can be visited. For the above examples, this pattern matches h_1, h_3, and h_4. Since there are two matches for h_4, the total number of matches, called the *support* of the pattern, is four.

A typical objective of sequential pattern mining is to enumerate all the frequent sequential patterns. A *frequent sequential pattern* is defined as a sequential pattern such that its support is greater than or equal to the given threshold, called *min-support*. If we take the domain-independent approach, namely, if we treat locations as symbols, we can directly apply sequential pattern mining to movement histories, but the spatio-temporal nature of moving objects is lost. In the following, we introduce some ideas that incorporate the semantics of moving objects to sequential pattern mining.

11.3.1 TrajPattern: Finding the Top-k Trajectory Patterns

TrajPattern was proposed by Yang and Hu [66] and is a framework inspired by sequential pattern mining. The algorithm tries to summarize trajectories into k prominent movement patterns considering imprecision and noise in trajectories.

In their approach, a trajectory has the form $t = \langle (l_1, \sigma_1), (l_2, \sigma_2), \ldots \rangle$, where l_i and σ_i are the expected location and the standard deviation of the mobile object at the ith snapshot. The matching probability $\Pr(p, t)$ between a pattern $p = \langle p_1, \ldots, p_n \rangle$ and a trajectory $t = \langle (l_1, \sigma_1), \ldots, (l_n, \sigma_n) \rangle$ considers the ambiguity of the trajectory (its definition is omitted here). The *match* measure is defined as follows:

$$\mathrm{match}(p, t) = \frac{\log \Pr(p, t)}{|n|}. \tag{11.3}$$

When a trajectory t whose length is longer than pattern p is given, the match measure is extended as follows:

$$\mathrm{match}(p, t) = \max_{\forall t' \subseteq t, |t'| = |p|} \mathrm{match}(p, t'), \tag{11.4}$$

where '$\subseteq$' represents the subsequence relationship. When a trajectory data set D is given, the match measure is further extended as:

$$\mathrm{match}(p, D) = \sum_{t \in D} \mathrm{match}(p, t). \tag{11.5}$$

The score is regarded as the expected number of occurrences of pattern p in D. Roughly speaking, TrajPattern finds the k patterns with the highest scores.

TrajPattern is based on the sequence mining approach, but clustering is also used. It first identifies short patterns with high scores and then tries to extend the patterns to longer patterns with high scores. To prune nonqualifying candidates, the algorithm utilizes the *min-max property*, which is a similar notion to the *apriori property* [1, 25, 56], but it is a weaker notion due to the definition of the match measure.

11.3.2 Mobility Rules: Consideration of Cell Topologies and Noises

Yavaş et al. [67] proposed a method for mining *mobility rules* from a movement history (a sequence of cell numbers) by extending sequential pattern mining. First, the method extracts *user mobility patterns* from the given sequence. A user mobility pattern means a frequent cell sequence. The algorithm tries to find meaningful user mobility patterns by extending the sequential pattern mining approach. The following two extensions are essential:

- A user mobility pattern is generated such that the pattern consists of *neighboring* cells considering possible movements of mobile objects. That is, the method takes the underlying cell topology into account.

- A movement trajectory often contains noise due to random movements and corruption. They, therefore, provide a *robust* support counting method that utilizes the notion of *string alignment* to enable flexible string matching.

Second, *mobility rules* are generated. On the basis of the user mobility patterns extracted in the previous step, the algorithm generates rules such as $\langle \mathtt{A}, \mathtt{C} \rangle \rightarrow \langle \mathtt{D}, \mathtt{B} \rangle$. This rule means that an object that moves from `A` to `C` will move from `D` to `B` with high support and confidence.

11.3.3 Frequent Mobility Pattern Mining from One Long Trajectory

Cao et al. [4] mine frequent spatio-temporal patterns from *one* long trajectory. An example of such data is a trace of a bus for a single day in a city. The method detects frequent sequential patterns without predefined segmentation of a trajectory. First, the algorithm simplifies the given trajectory data into a list of approximated line segments. Next, similar line segments are grouped to find frequent sequential patterns. Then, each segment contained in each sequential pattern is converted into a sequence of region IDs, where region means the area surrounding a segment. Finally, longer frequent sequential patterns are derived by combining the short sequential patterns. To accelerate the derivation step, they use a tree structure and an apriori-like pruning technique.

11.3.4 Other Work

The algorithm proposed by Peng and Chen [47] identifies movement patterns from movement log data. Given a *support threshold*, the algorithm tries to find long sequential patterns such as "`ABDEB`," each of which corresponds to a cell in a mobile network. The mined movement patterns are used to allocate data in appropriate mobile sites, so that moving objects can obtain data efficiently while they move in a mobile network.

11.4 FINDING OTHER INTERESTING PATTERNS

The former section focused on the extension of sequential pattern mining. This section introduces other mining approaches to find interesting patterns from moving object databases.

11.4.1 Spatio-temporal Association Rules

Association rule mining is one of the most popular topics in data mining [1, 25, 56]. An example of an *association rule* is $\{\mathrm{notebook}\} \Rightarrow \{\mathrm{pen}\}$. The rule indicates that if a notebook is purchased, it is likely that a pen is also purchased. So far, various association rule mining methods for large transaction data have been proposed.

Verhein and Chawla [61] extend the notion of association rules to the spatio-temporal context. They call such rules *spatio-temporal association rules*. The most

simple type of rule can be written as

$$R = (r_i, \Delta_i) \Rightarrow (r_j, \Delta_j), \tag{11.6}$$

where r_i, r_j are spatial regions and Δ_i, Δ_j are time intervals that satisfy $\Delta_i < \Delta_j$. The above rule says that objects appearing in r_i during time interval Δ_i will appear in region r_j during Δ_j. To find interesting rules, they propose the notion of *spatial support*:

$$\text{spatial\_support}(R) = \frac{\sigma((r_i, \Delta_i) \Rightarrow (r_j, \Delta_j))}{\text{area}(r_i) + \text{area}(r_j)}, \tag{11.7}$$

where $\sigma((r_i, \Delta_i) \Rightarrow (r_j, \Delta_j))$ denotes the conventional *support* of the rule—the number of objects that satisfy the rule and $\text{area}(r)$ is the area of region r. The smaller the area covered, the higher the spatial support of the rule.

In addition, several interesting patterns are proposed in Ref. [61]. Some examples are as follows:

- *Dense region*: A region r is called a *dense region* (or *hot spot*) during Δ if $\text{density}(r, \Delta) = \sigma(r, \Delta)/\text{area}(r) \geq \delta$, where $\sigma(r, \Delta)$ is the number of objects in r during Δ and δ is a *minimum density threshold*.

- *High traffic region*: A region r is a *high traffic region* if the number of objects entering r (n_r) or leaving r (n_l) during Δ satisfies $\alpha/\text{area}(r) \geq \tau$, where $\alpha = n_e$ or n_l and τ is a *minimum traffic threshold*.

- *Stationary region*: If $\sigma((r, \Delta_i) \Rightarrow (r, \Delta_{i+1}))/\text{area}(r) \geq \tau$, r is called a *stationary region.*

In Ref. [62], the methodology is further refined by one of the authors to represent longer patterns.

11.4.2 Group Patterns

The algorithm proposed by Wang et al. [60] discovers *groupings* of moving objects such that members in the same group are spatially close to one another for a significant amount of time. Such object groupings are called *group patterns*. Given movement histories of objects, the algorithm tries to find appropriate groupings based on the following criteria: (1) the group members should be physically close to one another and (2) the group members should stay together for some meaningful duration.

Two group pattern mining algorithms *apriori-like group pattern mining* (*AGP*) and *valid group-growth* (*VG-Growth*) are proposed. The former is an extension of the well-known *apriori* algorithm [1] and the latter is based on the *FP-growth* algorithm [24]. Some techniques for accelerating the mining process are also introduced.

11.4.3 Periodic Movement Patterns

Moving objects often follow *periodic movements*. For example, a bus run on a regular route shows similar movement patterns every day. Mamoulis et al. [39] try to discover *periodic movement patterns* in spatio-temporal data, including a long movement history of *one* moving object. The approach is an extension of *periodic pattern mining* from event sequences [23].

A *periodic pattern* is defined as a sequence of spatial regions that appears every T timestamp: the pattern should appear at least min_sup periodic intervals in the input trajectory. For example, "`AB*C*D`" is a pattern with length $T = 6$, where "`*`" is a "don't care" character and matches any region. Namely, the pattern means that the object visits regions from `A` to `D` in order and in a cyclic manner. An interesting aspect is that the regions are not predefined; the algorithm *discovers* appropriate regions to form movement patterns. Regions are defined as dense areas and determined using a method inspired by *density-based clustering* [25, 56]. The method is, therefore, prone to small amounts of noise in the trajectory.

Cao et al. [5] further extend the idea and propose variations of periodic patterns and ways to discover them.

11.4.4 Flock, Leadership, Convergence, and Encounter

The *REMO* (RElative MOtion) framework [34, 35] developed by Laube et al. defines various types of behavior pattern for moving object groups. Gudmundsson et al. [14, 15] select some patterns from the framework and then provide formalized definitions:

- *Flock*: At least n objects are within a circle of radius r and they move in the same direction.
- *Leadership*: In addition to the flock pattern condition, the object group should satisfy an additional condition: one of the objects is heading the direction for at least τ time steps.
- *Convergence*: At least n objects pass through the same circle of radius r without changing direction, but the objects need not arrive at the same time.
- *Encounter*: This is a specialization of the convergence pattern. At least n objects are simultaneously inside the same circle of radius r.

References [14, 15] provide efficient computation algorithms in terms of computational geometry using approximation techniques.

11.4.5 More Complex Patterns

Some other researchers have proposed using more complex patterns to represent complex behaviors of moving objects. Although their objectives are data representa-

tion and query processing, the ideas may be applied to data mining on moving object databases.

Mobility patterns, as proposed by du Mouza and Rigaux [43], is a language to represent movement patterns between location areas. The following are examples of mobility patterns:

- Give all the objects that travel from `A` to `F` and from `F` to `C` in 10 minutes: `start_at A, follow F, roam 10, follow C`
- Give all the objects that went through `F` to another area, then went to `D` or `C`, and came back to `F` using the same area: `follow F.@x, follow {D, C}, follow @x.F; @x != F`

Symbols such as `A` are labels for areas and `@x` is a variable. In Ref. [42], this idea is generalized to trajectories at multiple resolutions. The target space can be represented at different levels of coarseness and mobility patterns are generalized depending on levels.

Hadjieleftheriou et al. [19] also proposed the notion of complex spatio-temporal pattern queries with their efficient processing methods. Two types of spatio-temporal queries are considered:

- *Spatio-temporal queries with time*: Arbitrary types of spatial predicate may be contained (e.g., range search), and each predicate can be associated with an exact temporal constraint. An example is "find objects that crossed through region `A` at time T_1, came as close as possible to point `B` at a later time T_2 and then stopped inside circle `C` some time during interval (T_3, T_4)."
- *Spatio-temporal queries with order*: The difference with respect to the former is that each predicate is associated with a relative order. In this sense, they are more general than the former. An example is "find objects that first crossed through region `A`, then passed as close as possible from point `B` and finally stopped inside circle `C`."

Efficient query processing methods that use indexing methods have been proposed.

Jin et al. [29] tried to find movement patterns from user movement logs. They considered graph structures in which nodes correspond to locations. Their algorithm considers support counts of node traversals and finds typical movement patterns. An interesting point is that the algorithm finds seldom-visited nodes and random walks from movement logs. A *random walk* consists of multiple nodes that the user moves between frequently in a random manner. Location prediction and location query techniques based on the mined movement behaviors are also proposed.

11.5 CLUSTERING MOVING OBJECTS

Clustering is a technique for grouping a large number of objects and generates *clusters*, which are used to summarize the original dataset. There have been a lot

of clustering algorithms proposed in data mining [25, 56]. In the following, we introduce some clustering techniques for moving object databases.

11.5.1 Continual Maintenance of Moving Clusters

Li et al. [36] proposed a real-time and adaptive cluster maintenance method for moving points. The approach is based on the notion of micro-clusters. A *micro-cluster* is a small-sized cluster consisting of nearby objects. After the generation of micro-clusters, some different clustering algorithms can be applied to the micro-clusters by treating each micro-cluster as if it were an individual entity. The idea of micro-clusters was initially proposed in *BIRCH* [69]. The method in Ref. [36] generates *moving micro-clusters* from the target moving objects, and then global clusters are generated using the micro-clusters. Since moving objects change positions and directions, the method maintains clusters adaptively.

The merging and partitioning processes for micro-clusters are performed using *clustering features*, which summarize the clusters. A clustering feature for a (micro-) cluster C_i is defined as

$$cf_i = (sx_i, sy_i, sv_{x_i}, sv_{y_i}, n_i, t_i), \tag{11.8}$$

where t_i is the cluster generation time, n_i the number of elements ($n_i = |C_i|$), and sx_i (sy_i) the sum of the x-axis (y-axis) values of the elements ($sx_i = \sum_{j:o_j \in C_i} x_j$). sv_{x_i} (sv_{y_i}) is the sum of x-axis (y-axis) velocity values for the elements ($sv_{x_i} = \sum_{j:o_j \in C_i} v_{x_j}$).

When two clusters C_i, C_j are merged at time t_k ($t_i, t_j < t_k$), the clustering feature cf_k of the result cluster C_k is defined as

$$cf_k = (sx_k, sy_k, sv_{x_i} + sv_{x_j}, sv_{y_i} + sv_{y_j}, n_i + n_j, t), \tag{11.9}$$

where sx_k is defined as follows:

$$sx_k = sx_i + (t - t_i)sv_{x_i} + sx_j + (t - t_j)sv_{x_j}. \tag{11.10}$$

sy_k is defined in a similar manner. When partitioning a cluster into two clusters, the resulting clustering features can also be easily computed (the calculation method is omitted here).

An interesting point of this approach is its cluster management scheme. It tries to keep the spatial extent of moving micro-clusters small. The compactness of a micro-cluster is measured by its bounding rectangle. If the size of the bounding rectangle exceeds a certain threshold, the micro-cluster is split.

11.5.2 Detecting Moving Clusters from Object Movement Histories

Kalnis et al. [30] identify moving clusters in long movement histories . Intuitively, a *moving cluster* in their approach is a sequence of spatial clusters that appear in

consecutive snapshots of object movements, such as two consecutive spatial clusters that share a large number of common objects.

The basic idea is as follows. The input is the snapshots of moving object positions. If two clusters C_t at time t and and C_{t+1} at time $t+1$ satisfy the condition

$$\frac{|C_t \cap C_{t+1}|}{|C_t \cup C_{t+1}|} \geq \theta, \tag{11.11}$$

(C_t, C_{t+1}) is called a *moving cluster*. The idea resembles the notion of dense regions which will be described in Section 11.6, but the difference is that clusters move continually while sharing objects. A number of cluster-detection algorithms have been proposed.

Figure 11.4 shows an example illustrating the concept. S_t, S_{t+1}, and S_{t+2} are three consecutive snapshots of movements. Each circle in each snapshot represents a cluster. If a threshold value, say, $\theta = 0.5$, is used, then three clusters are treated as one moving cluster.

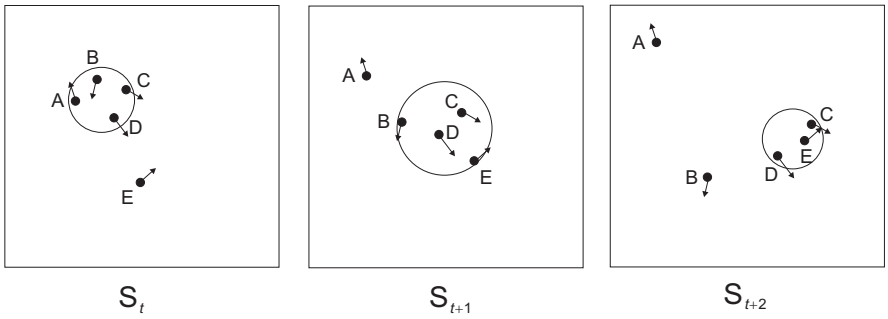

Fig. 11.4 Example of a moving cluster

11.5.3 Clustering Moving Objects while Considering Positional Uncertainty

The clustering method proposed by Kriegel and Pfeifle [31] considers *uncertainty* in the positions of moving objects. Their focus is the fuzzy nature of the positions of moving objects. The uncertainty of a moving object is modeled by a *spatial density function* that represents the likelihood that a certain object is located at a certain position. Since the locations of moving objects are uncertain, their algorithm performs several clusterings of points sampled from the probability density functions of the moving objects. A ranking value is assigned to each of the obtained clusterings, which reflects its distance to other sample clusterings. The clustering with the smallest ranking value is called the *medoid clustering* and can be regarded as the average clustering of all the sample clusterings. The method obtains robust clustering results based on this approach.

11.5.4 Other Work

Nanni and Pedreschi [44] employed *density-based clustering* [25, 56] to cluster trajectories. This method groups objects into clusters based on *density*, which is the population within a given region in the space. Typical constraints used in density-based clustering are as follows: for each object in a cluster, its neighborhood, defined by a given radius ε, must contain at least a minimum number of objects, n. Density-based clustering has some advantageous features: it can detect nonspherical clusters of arbitrary shapes and it is robust with respect to noise. Since trajectories may have "snake" shapes and often contain noise, such features are desirable.

Yiu and Mamoulis [68] presented a method to cluster objects on a *spatial network* such as a road network. Each object (not necessarily a moving object) lies on an edge of a large network. The distance between objects is defined by the length of the shortest path between them over the network. Variants of partitioning, density-based, and hierarchical clustering methods have been developed.

The method proposed by Zhang and Lin [71] generates k clusters considering the positions, speeds, and sizes of moving objects. They proposed a special distance function that considers speeds and positions and formalized the clustering as a k-*center optimization* problem. An approximation-based efficient solution method and a cluster refinement method were proposed. The constructed clusters are used to estimate the selectivity of queries on a moving object database.

There exist statistical model-based approaches for trajectory clustering. Gaffney and Smyth [13] applied regression models to cluster similar trajectories. The approach considers two trajectories to be "similar" when they are likely to be generated from a common core trajectory by adding Gaussian noise. A clustering method that is invariant to spatial and temporal shifting of trajectories within clusters was also proposed [10]. The technique was applied to the clustering of human motion trajectories, cyclone trajectories, and so on.

There are some approaches to clustering moving objects from a theoretical perspective. Hershberger [27] proposed a deterministic kinetic data structure for maintaining a covering of moving points in R^d by d-dimensional boxes in an online fashion. The number of boxes is always within a factor of 3^d of the best possible static covering. Har-Peled [26] shows how to partition n linearly moving points into k^2 static clusters, so that at any time the diameter of each cluster is at most equal to the maximum cluster diameter in an optimal k-clustering for the current point positions. However, all the movements must be known in advance.

11.6 DENSE REGIONS AND SELECTIVITY ESTIMATION

A region on a space is called a *dense region* if the number of moving objects contained in the region is above some threshold. Detection of dense regions from the underlying moving object database is highly related to density-based clustering (described above). Dense region detection is also related to the estimation problem regarding

query selectivity for moving object databases. Some dense region detection methods and query selectivity estimation techniques are briefly reviewed below.

11.6.1 Detecting Dense Regions

Hadjieleftheriou et al. [17] considered processing *density-based queries* on moving object databases. The *density* of region r during time interval Δt is defined as:

$$\text{density}(r, \Delta t) = \frac{\min_{t \in \Delta t} n(r, t)}{\text{area}(r)}, \tag{11.12}$$

where $n(r, t)$ is the number of objects inside r at time t and $\text{area}(r)$ is the area of r. This definition of a dense region is intuitive, but tiny dense regions are also detected. To detect meaningful dense regions, they extend the above basic notion.

For example, a *period density query* is defined as follows. Given movement trajectories, a constant H, and thresholds α_1, α_2, and ξ, find regions $\{r_1, \ldots, r_k\}$ and associated maximal time intervals $\{\Delta t_1, \ldots, \Delta t_k \mid \Delta t_i \subset [t_{\text{now}}, t_{\text{now}} + H]\}$ such that $\alpha_1 \leq \text{area}(r_i) \leq \alpha_2$ and $\text{density}(r_i, \Delta t_i) > \xi$, where t_{now} is the current time. Some algorithms have been provided to find dense areas from a moving object database of linear movements with uniform speeds. To simplify the problem, they partition the data space into disjoint cells instead of arbitrary regions and then find dense regions.

Jensen et al. [28] focused on the identification of dense regions at time $t \in [t_{\text{now}}, t_{\text{now}} + H]$. Objects move continuously and their positions and velocities are updated often. Queries are processed in an online setting, where they assume that objects move linearly until changes are reported. The algorithm computes given queries efficiently using a *density histogram*, which is maintained online.

11.6.2 Selectivity Estimation and Histograms

The *selectivity* of a query is a ratio indicating how many of the objects in the database satisfy the given query [48]. Assume that a query to a moving object database such as "retrieve all the objects that enter a specified region r at time t" is given. To construct an efficient query evaluation plan, the moving object database system needs to estimate the number of objects that qualify for the query. Several methods have been proposed to estimate spatio-temporal query selectivitiy. A typical approach to estimating query selectivity is to construct a *histogram* [20], which is a compact structure summarizing the underlying database statistics.

11.6.2.1 Histogram for Static Queries on Moving Objects

Choi and Chung [8] extended the traditional histogram technique for spatial databases to cope with linearly moving point objects. The method estimates the selectivity of spatial range queries. Given a rectangular range r and timestamp t that represents some future time, the method estimates the selectivity. In this sense, the method

focuses on the case of moving points and static queries. To create a histogram, moving objects are clustered based on their current positions and then organized into buckets. For each bucket, a *spatial bounding rectangle* that covers the objects within the bucket and a *velocity bounding rectangle* that bounds the velocities of the objects within the bucket are constructed. However, the histogram should be rebuilt frequently for accurate estimation. Estimation is performed by assuming the uniformity of velocities. The same authors proposed a further improved method [9]. Nonuniformity of velocities is also considered and a refined histogram is constructed.

Hadjieleftheriou et al. [18] also propose a selectivity estimation method in which they assume linear trajectories. In their approach, the *duality transform* technique is used: the moving points in the primal space-time space are transformed into dual velocity-intercept space. A histogram is then constructed on the dual velocity-intercept space.

11.6.2.2 Histogram for Moving Queries on Moving Objects

In contrast, Tao et al. [57] proposed a further improved method. Their multidimensional *spatio-temporal histogram* supports all types of objects (static/dynamic and points/rectangles) and moving queries. Their method constructs a spatio-temporal histogram, which considers both locations and velocities for partitioning. In addition, an incremental histogram maintenance method was proposed.

11.6.2.3 Other Work

The approach of Sun et al. [55] is to use an adaptive multidimensional histogram to summarize the positions of moving objects for the present time. Past histograms are archived as *historical synopses* and allow users to issue aggregate queries related to the past. In addition, a prediction method is proposed for future movements based on the current movement statistics. Tao et al. [58] present an interesting approach. The method integrates spatio-temporal indexes with sketches in order to aggregate spatio-temporal statistics, including object movements. A *sketch* is a common approach to approximate counting information and is considered as a special kind of histogram. The idea proposed is applicable for finding spatio-temporal association rules such as $(r_i, T, p) \Rightarrow r_j$. This rule means that a user in region r_i at time t will appear in region r_j by time $t + T$ with probability p. Other approaches to spatio-temporal histograms for moving objects can be found in Refs. [12, 46].

11.6.3 Mobility Histograms Based on Markov Chains

Our mobility representation model using Markov chains over spatial grid cells is presented in subsection 11.2.2. In this subsection, we describe its histogram structure for summarizing mobility statistics [22]. The histogram structure has two representation levels: logical and physical.

11.6.3.1 Logical Level: Data Cubes

To represent order-k Markov chain-based movement statistics, a *mobility histogram* is constructed as a $(k+1)$-dimensional data cube. A *data cube* [25] is a data structure that summarizes the underlying data in a multidimensional array and is often used in OLAP, which provides flexible analysis facilities. Figure 11.5 shows an example of data cube for $k = 2$. The data cube corresponds to the level-1 space partitioning ($P = 1$), mentioned in subsection 11.2.2. Since the two-dimensional target space is partitioned into $R = 2^{2P} = 4$ spatial regions, the data cube contains $R^{k+1} = 64$ cells. For each dimension of the data cube, each step 0, 1, and 2 corresponds to each step of an order-2 Markov chain. For instance, when the sequence $1^{(1)} \rightarrow 1^{(1)} \rightarrow 2^{(1)}$ is given as an input transition sequence, the value of the corresponding cell $(1, 1, 2)$ is incremented.

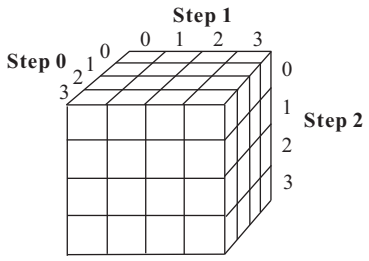

Fig. 11.5 Logical histogram representation as a data cube

More intuitive data manipulation is possible using the data cube representation. For example, consider the data cube shown in Figure 11.5. The probability that an object has moved from region 1 to region 2 and then moves to region 4 is calculated as $\Pr(4|1, 2) = \mathrm{val}(1, 2, 4)/\mathrm{val}(1, 2, *)$, where $\mathrm{val}(1, 2, 4)$ is the value of the cube cell $(1, 2, 4)$ and $\mathrm{val}(1, 2, *) = \sum_{i=0}^{2^P-1} \mathrm{val}(1, 2, i)$. Moreover, the probability that an object in region 1 at $t = \tau$ and in region 3 at $t = \tau + 2$ is in region 2 at $t = \tau + 1$ (τ is some arbitrary time) is calculated as $\mathrm{val}(1, 2, 3)/\mathrm{val}(1, *, 3)$.

As described in subsection 11.2.2, our mobility model allows multiple resolutions using different partition level settings. Roll-up and drill-down operations are also supported for data cubes and users can change resolutions when they perform analyses. Other types of queries can also be supported by the data cube representation. For example, density queries can be processed by a data cube using aggregation and selection.

11.6.3.2 Physical Histogram: Multidimensional Trie

A physical histogram has a structure like a *multidimensional trie* for summarizing mobility statistics with multiple resolutions. It has the following features:

1. Each node of a trie has four branches labeled `00`, `01`, `10`, and `11`. Each of the branches corresponds to a quarter region obtained by decomposing the space

into 2×2 components. That is, the space decomposition is performed in a similar manner to a *quadtree* [51].

2. The first branch of the trie root corresponds to step 0 of a Markov chain and the next branch to step 1, and so on. Each branch thus corresponds to a Markov transition steps in turn. The idea is inherited from *k-d trees* [51].

3. Each node of a trie has a *counter* for accumulating the number of trajectory patterns visited.

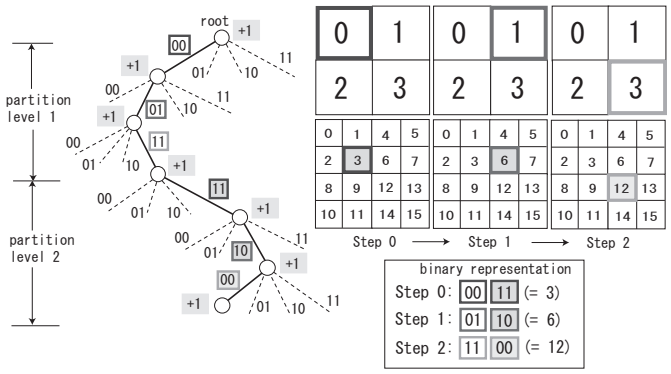

Fig. 11.6 Physical histogram structure based on a multidimensional trie.

Figure 11.6 shows the structure in the case of the maximal partition level $P = 2$. The dotted lines mean that the edges are not instantiated because the corresponding sequences do not appear. The example shows the situation that a transition sequence $3^{(2)} \rightarrow 6^{(2)} \rightarrow 12^{(2)}$ is inserted. As described in subsection 11.2.2, the model assigns region numbers based on the Z-ordering method. The merit of using this numbering method is that the four-way branching approach corresponds exactly to the Z-ordering numbering. Using this approach, we can accumulate movement patterns adaptively so as not to create unnecessary branches for the trie. Furthermore, in order to reduce the storage overhead of the histogram, an approximate histogram construction algorithm was proposed [22]. The algorithm receives movement trajectories as a stream and gradually expands the trie considering the statistical properties of the nodes in the trie. Although the histogram constructed does not contain exact count information, it represents the overall statistics approximately with a relatively small storage cost.

11.7 COMPARING MOVING OBJECT TRAJECTORIES

11.7.1 Distance Between Trajectories

To perform clustering or similarity searches on trajectories of moving objects, an appropriate definition of the *distance* (or similarity) between two trajectories is important. Typically, the trajectory of a moving object is represented as a sequence of consecutive locations in a two- or three- dimensional space, in contrast to the one-dimensional case, which is common in timeseries databases [33, 52]. In addition, distances for moving object trajectories should be robust to outliers since measurements in mobile environments are noisy and the movement of objects may contain "gaps." Some distance measures proposed for moving object trajectories are investigated below.

For example, consider two trajectories of moving objects on the (x, y)-plane given as $A = ((a_{x,1}, a_{y,1}), \ldots, (a_{x,n}, a_{y,n}))$ and $B = ((b_{x,1}, b_{y,1}), \ldots, (b_{x,m}, b_{y,m}))$. Their lengths are n and m, respectively. When $n = m$, we can apply the *Euclidean distance* (L_2 distance), which is simple and also the most popular:

$$L_2(A, B) = \left(\sum_{i=1}^{n} \left[(a_{x,i} - b_{x,i})^2 + (a_{y,i} - b_{y,i})^2 \right] \right)^{\frac{1}{2}}. \tag{11.13}$$

Although the Euclidean distance can be evaluated efficiently, it cannot be applied when two trajectories have different lengths.

11.7.2 Dynamic Time Warping (DTW)

It is often necessary to compute the similarity between two trajectories with different lengths. A well-known approach is to use *dynamic time warping* (*DTW*) [33, 52], which is defined as follows:

$$\begin{aligned} \mathrm{DTW}(A, B) \quad = \quad & |a_n, b_m| + \min\{\mathrm{DTW}(\mathrm{head}(A), \mathrm{head}(B)), \\ & \mathrm{DTW}(\mathrm{head}(A), B), \mathrm{DTW}(A, \mathrm{head}(B))\}, \end{aligned} \tag{11.14}$$

where $|a_n, b_m|$ is the distance between the two points $a_n = (a_{x,n}, a_{y,n})$ and $b_n = (b_{x,m}, b_{y,m})$ and is usually measured using the Euclidean distance. $\mathrm{head}(A)$ is a sequence A without the last item $(a_{x,n}, a_{y,n})$. An example of DTW matching is shown in Figure 11.7, where A and B are two trajectories. DTW is the accumulated distance between the matching nodes and allows flexible sequence matching, but it is not an effective distance metric for noisy trajectory data because all the elements in two trajectories must be matched when using DTW. In the figure, three points x, y, and z are outliers, but DTW tries to match these points. Although the two trajectories are similar except for these outliers, DTW returns a large distance value.

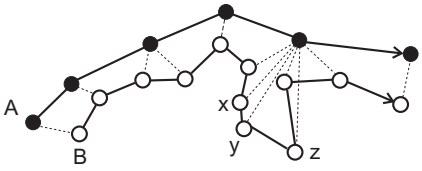

Fig. 11.7 DTW matching.

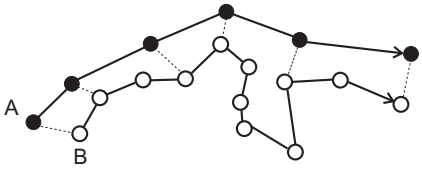

Fig. 11.8 LCSS matching.

11.7.3 More Robust Distance Measures

On the basis of this observation, Vlachos et al. [63] proposed the *least common subsequence* (*LCSS*) distance for similarity-based retrieval of moving object trajectories. The LCSS score is defined as follows (we have simplified the original definition for simplicity):

$$\mathrm{LCSS}(A,B) = \begin{cases} 0 & \text{if } A \text{ or } B \text{ is empty} \\ 1 + \mathrm{LCSS}(\mathrm{head}(A), \mathrm{head}(B)) & \\ \quad \text{if } |a_n - b_m| < \varepsilon \text{ and } |n-m| \leq \delta & \\ \max(\mathrm{LCSS}(\mathrm{head}(A), B), \mathrm{LCSS}(A, \mathrm{head}(B))) & \\ \quad \text{otherwise} & \end{cases} \tag{11.15}$$

The parameters ε and δ are given by the user. The condition $|a_n - b_m| < \varepsilon$ means that a_n and b_m can be regarded as if they are the *same symbols*. The LCSS score means the number of matching symbols and takes a large value when two trajectories are similar. Figure 11.8 shows an example of LCSS matching where $\mathrm{LCSS}(A,B) = 6$. Vlachos et al. defined the *LCSS distance* with the following formula:

$$D(A,B) = 1 - \frac{\mathrm{LCSS}(A,B)}{\min(n,m)}. \tag{11.16}$$

As shown in the figure, the LCSS matching omits outliers so that it is more robust to noise. They extend the distance considering time stretching and translations.

Chen et al. [7] further extended the idea in this direction. They proposed a new distance called *edit distance on real sequence* (*EDR*) for moving object trajectories. The basic idea of EDR is that it assigns penalties to nonmatched points. EDR is therefore more sensitive than LCSS according to the number of outliers. The distance between two trajectories becomes small when they are similar in the sense of LCSS and have less outliers. This distance function was shown to be more robust than DTW and LCSS over trajectories with noise.

11.7.4 Other Work

Yanagisawa et al. [64] discussed the issues of shape-based similarity queries for moving object trajectories. They proposed some similarity measures considering

trajectory approximation. Yanagisawa and Satoh [65] define two distance measures for trajectories. They are extensions of the Euclidean distance and DTW respectively and consider the shapes and velocities of moving object trajectories. Lin and Su [37] proposed the "one way distance" function for comparing moving object trajectories. In addition, several techniques for implementation were presented.

11.8 CONCLUSIONS

In this chapter, we have reviewed current trends in data mining technologies on moving object databases. Data mining on moving object databases has different requirements from conventional data mining since moving objects have a dynamic nature and spatio-temporal semantics, and different use is made of mined knowledge. The new requirements give birth to new technologies. As described above, a variety of interesting approaches have appeared in this field of research.

We have also shown some pointers to related areas. The *spatio-temporal database* technology is a closely related topic in moving object databases. It is a generic name for databases that store and manage information regarding objects with temporal and spatial features, and includes the notion of moving objects databases. A spatio-temporal database is, however, not necessarily for moving objects because it is used, for example, for representation of time-varying geographic information. Refs. [32, 41] are collections of articles on spatio-temporal data mining. Roddick et al. [50] provides a reference list concerning data mining technology including spatio-temporal databases up to the year 2000. López et al. [38] survey aggregation techniques for spatial, temporal, and spatio-temporal databases. Aggregation is used to accumulate statistics from an underlying database and is useful for data mining and learning from the data. Dunham et al. [11] and Wang et al. [59] review spatio-temporal data mining technologies. There are a number of textbooks on spatial databases [40, 49, 51, 53], and well-known data mining textbooks [25, 56].

Acknowledgments

This research is partly supported by the Grant-in-Aid for Scientific Research on Priority Areas (19024037, 21013023) from the Ministry of Education, Culture, Sports, Science and Technology (MEXT, Japan and the Grant-in-Aid for Scientific Research (19300027) from Japan Society for Promotion of Science (JSPS).

REFERENCES

1. R. Agrawal and R. Srikant. Fast algorithms for mining association rules in large databases. *Proc. Int'l Conf. on Very Large Data Bases (VLDB'94)*, pp. 487–499, 1994.

2. R. Agrawal and R. Srikant. Mining sequential patterns. *Proc. Int'l Conf. on Data Engineering (ICDE'95)*, pp. 3–14, 1995.

3. A. Bhattacharya and S.K. Das. LeZi-Update: An information-theoretic framework for personal mobility tracking in PCS networks. *ACM/Kluwer Wireless Networks*, 8(2-3), pp. 121–135, 2002.

4. H. Cao, N. Mamoulis, and D.W. Cheung. Mining frequent spatio-temporal sequential patterns. *Proc. Int'l Conf. on Data Mining (ICDM'05)*, pp. 82–89, 2005.

5. H. Cao, N. Mamoulis, and D.W. Cheung. Discovery of periodic patterns in spatiotemporal sequences. *IEEE Trans. on Knowledge and Data Engineering*, 19(4):453–467, 2007.

6. C. Cheng, R. Jain, E. van den Berg. Location prediction algorithms for mobile wireless systems. In B. Furht and M. Ilyas (eds.), *Wireless Internet Handbook: Technologies, Standards, and Applications*, CRC Press, pp. 245–263, 2003.

7. L. Chen, M.T. Özsu, and V. Oria. Robust and fast similarity search for moving object trajectories. *Proc. ACM SIGMOD Int'l Conf. on Management of Data (SIGMOD'05)*, pp. 491–502, 2005.

8. Y.-J. Choi and C.-W. Chung. Selectivity estimation for spatio-temporal queries for moving objects. *Proc. ACM SIGMOD Int'l Conf. on Management of Data (SIGMOD'02)*, pp. 440–451, 2002.

9. Y.-J. Choi, H.-H. Park, and C.-W. Chung. Estimating the result size of a query to velocity skewed moving objects. *Information Processing Letters*, 88(6), pp. 279–285, 2003.

10. D. Chudova, S. Gaffney, E. Mjolsness, and P. Smyth. Translation-invariant mixture models for curve clustering. *Proc. ACM SIGKDD Int'l Conf. on Knowledge Discovery and Data Mining (KDD'03)*, pp. 79–88, 2003.

11. M.H. Dunham, N. Ayewah, Z. Li, K. Bean, and J. Huang. Spatio-temporal prediction using data mining tools. In [40].

12. H.G. Elmongui, M.F. Mokbel, and W.G. Aref. Spatio-temporal histograms. *Proc. Int'l Symp. on Spatial and Temporal Databases (SSTD'03)*, LNCS 3633, pp. 19–36, 2005.

13. S. Gaffney and P. Smyth. Trajectory clustering with mixtures of regression models. *Proc. ACM SIGKDD Int'l Conf. on Knowledge Discovery and Data Mining (KDD'99)*, pp. 63–72, 1999.

14. J. Gudmundsson, M. van Kreveld, and B. Speckmann. Efficient detection of motion patterns in spatio-temporal data sets. *Proc. ACM Int'l Workshop on Geographic Information Systems (GIS'04)*, pp. 250–257, 2004.

15. J. Gudmundsson and M. van Kreveld. Computing longest duration flocks in trajectory data. *Proc. ACM Int'l Workshop on Geographic Information Systems (GIS'06)*, pp. 35–42, 2006.

16. R.H. Güting and M. Schneider. *Moving Objects Databases*. Morgan Kaufmann, 2005.

17. M. Hadjieleftheriou, G. Kollios, D. Gunopulos, and V.J. Tsotras. On-line discovery of dense areas in spatio-temporal databases. *Proc. Int'l Symp. on Spatial and Temporal Databases (SSTD'03)*, pp. 306–324, 2003.

18. M. Hadjieleftheriou, G. Kollios, and V.J. Tsotras. Performance evaluation of spatio-temporal selectivity estimation techniques. *Proc. Int'l Conf. on Scientific and Statistical Database Management (SSDBM'03)*, pp. 202–211, 2003.

19. M. Hadjieleftheriou, G. Kollios, P. Bakalov, and V.J. Tsotras. Complex spatio-temporal pattern queries. *Proc. Int'l Conf. on Very Large Data Bases (VLDB'05)*, pp. 877–888, 2005.

20. Y. Ioannidis. The history of histograms (abridged). *Proc. Int'l Conf. on Very Large Data Bases (VLDB'03)*, pp. 19–30, 2003.

21. Y. Ishikawa, Y. Tsukamoto, and H. Kitagawa. Extracting mobility statistics from indexed spatio-temporal databases *Proc. Workshop on Spatio-Temporal Database Management (STDBM'04)*, pp. 9–16, 1004.

22. Y. Ishikawa, Y. Machida, and H. Kitagawa. A dynamic mobility histogram construction method based on Markov chains. *Proc. Int'l Conf. on Scientific and Statistical Database Management (SSDBM'06)*, pp. 359–368, 2006.

23. J. Han, G. Dong, and Y. Yin. Efficient mining of partial periodic patterns in time series database. *Proc. Int'l Conf. on Data Engineering (ICDE'99)*, pp. 106–115, 1999.

24. J. Han, J. Pei, and Y. Yin. Mining frequent patterns without candidate generation. *Proc. ACM SIGMOD Int'l Conf. on Management of Data (SIGMOD'00)*, pp. 1–12, 2000.

25. J. Han and M. Kamber. *Data Mining*. Morgan Kaufmann, 2nd edition, 2005.

26. S. Har-Peled. Clustering motion. *Discrete & Computational Geometry*, 31(4), pp. 545–565, 2004.

27. J. Hershberger. Smooth kinetic maintenance of clusters. *Computational Geometry*, 31(1-2):3–30, 2005.

28. C.S. Jensen, D. Lin, B.C. Ooi, and R. Zhang. Effective density queries on continuously moving objects. *Proc. Int'l Conf. on Data Engineering (ICDE'06)*, 2006.

29. M.-H. Jin, J.-T. Horng, M.-F. Tsai, and E.H.-K. Wu. Location query based on moving behaviors. *Information Systems*, 32(3):385–401, 2007.

30. P. Kalnis, N. Mamoulis, and S. Bakiras. On discovering moving clusters in spatio-temporal data. *Proc. Int'l Symp. on Spatial and Temporal Databases (SSTD'05)*, LNCS 3633, pp. 364–381, 2005.

31. H.-P. Kriegel and M. Pfeifle. Clustering moving objects via medoid clusterings. *Proc. Int'l Conf. on Scientific and Statistical Database Management (SSDBM'05)*, 153–162, 2005.

32. R. Ladner, K. Shaw, and M. Abdelguerfi (eds.). *Mining Spatio-Temporal Information Systems*. Kluwer, 2002.

33. M. Last, A. Kandel, and H. Bunke (eds.) *Data Mining in Time Series Databases*. World Scientific, 2004.

34. P. Laube and S. Imfeld. Analyzing relative motion within groups of trackable moving point objects. *Proc. Int'l Conf. on Geographic Information Science (GIScience'02)*, LNCS 2478, pp. 132–144, 2002.

35. P. Laube, M. van Kreveld, and S. Imfeld. Finding REMO - Detecting relative motion patterns in geospatial lifelines. *Proc. Symp. on Spatial Data Handling (SDH'04)*, pp. 201–215, 2004.

36. Y. Li, J. Han, and J. Yang. Clustering moving objects. *Proc. ACM SIGKDD Int'l Conf. on Knowledge Discovery and Data Mining (KDD'04)*, pp. 617–622, 2004.

37. B. Lin and J. Su. Shapes based trajectory queries for moving objects. *Proc. ACM Int'l Workshop on Geographic Information Systems (GIS'05)*, pp. 21–30, 2005.

38. I.F.V. López, R.T. Snodgrass, and B. Moon. Spatiotemporal aggregate computation: A survey. *IEEE Trans. on Knowledge and Data Engineering*, 17(2):271–286, 2005.

39. N. Mamoulis, H. Cao, G. Kollios, M. Hadjieleftheriou, Y. Tao, and D.W. Cheung. Mining, indexing, and querying historical spatiotemporal data. *Proc. ACM SIGKDD Int'l Conf. on Knowledge Discovery and Data Mining (KDD'04)*, pp. 236–245, 2004.

40. Y. Manolopoulos, A.N. Papadopoulos, and M.G. Vassilakopoulos. *Spatial Databases: Technologies, Techniques and Trends*. Idea Group Publishing, 2004.

41. H.J. Miller and J. Han (eds.) *Geographic Data Mining and Knowledge Discovery*. Taylor & Francis, 2001.

42. C. du Mouza and P. Rigaux. Multi-scale classification of moving object trajectories. *Proc. Int'l Conf. on Scientific and Statistical Database Management (SSDBM'04)*, pp. 307–316, 2004.

43. C. du Mouza and P. Rigaux. Mobility patterns. *GeoInformatica*, 9(4):297–319, 2005.

44. M. Nanni and D. Pedreschi. Time-focused clustering of trajectories of moving objects. *Journal of Intelligent Information Systems*, 27(3):267–289, 2006.

45. M. Nelson and J.-L. Gailly. *The Data Compression Book*. M&T Books, New York, NY, 2nd edition, 1995.

46. H.K. Park, J.H. Son, and M.H. Kim. Dynamic histograms for future spatiotemporal range predicates. *Information Sciences*, 172(1–2), pp. 195–214, 2005.

47. W.-C. Peng and M.-S. Chen. Developing data allocation schemes by incremental mining of user moving patterns in a mobile computing system. *IEEE Trans. on Knowledge and Data Engineering*, 15(1), pp. 70–85, 2003.

48. R. Ramakrishnan and J. Gehrke. *Database Management Systems*. McGraw-Hill, 3rd edition, 2002.

49. P. Rigaux, M. Scholl, and A. Voisard. *Spatial Databases: With Application to GIS*. Morgan Kaufmann, 2001.

50. J.F. Roddick, K. Hornsby, and M. Spiliopoulou. An updated bibliography of temporal, spatial, and spatio-temporal data mining research. *Proc. Int'l Workshop on Temporal, Spatial, and Spatio-Temporal Data Mining (TSDM'00)*, LNCS 2007, pp. 147–164, 2000.

51. H. Samet. *Foundations of Multidimensional and Metric Data Structures*. Morgan Kaufmann, 2006.

52. D. Shasha and Y. Zhu. *High Performance Discovery in Time Series*. Springer-Verlag, 2004.

53. S. Shekhar and S. Chawla. *Spatial Databases: A Tour*. Prentice Hall, 2002.

54. L. Song, D. Kotz, and R. Jain. Evaluating next-cell predictors with extensive Wi-Fi mobility data. *IEEE Trans. on Mobile Computing*, 5(12):1633–1649, 2006.

55. J. Sun, D. Papadias, Y. Tao, and B. Liu. Querying about the past, the present, and the future in spatio-temporal databases. *Proc. Int'l Conf. on Data Engineering (ICDE'04)*, pp. 202–213, 2004.

56. P.-N. Tan, M. Steinbach, and V. Kumar. *Introduction to Data Mining*. Addison-Wesley, 2005.

57. Y. Tao, J. Sun, and D. Papadias. Selectivity estimation for predictive spatio-temporal queries. *Proc. Int'l Conf. on Data Engineering (ICDE'03)*, pp. 417–428, 2003.

58. Y. Tao, G. Kollios, J. Considine, F. Li, and D. Papadias. Spatio-temporal aggregation using sketches. *Proc. Int'l Conf. on Data Engineering (ICDE'04)*, pp. 214–226, 2004.

59. J. Wang, W. Hsu, and M.L. Lee. Mining in spatio-temporal databases. In [40].

60. Y. Wang, E.-P. Lim, and S.-Y. Hwang. Efficient mining of group patterns from user movement data. *Data & Knowledge Engineering*, 57(3):240–282, 2006.

61. F. Verhein and S. Chawla. Mining spatio-temporal association rules, sources, sinks, stationary regions and throughfares in object mobility databases. *Proc. Int'l Conf. on Database Systems for Advanced Applications (DASFAA'06)*, LNCS 3882, pp. 187–201, 2006.

62. F. Verhein. k-STARs: Sequences of spatio-temporal association rules. *Proc. Workshop on Spatial and Spatio-temporal Data Mining (SSTDM'06)*, 2006.

63. M. Vlachos, G. Kollios, and D. Gunopulos. Discovering similar multidimensional trajectories. *Proc. Int'l Conf. on Data Engineering (ICDE'02)*, pp. 673–684, 2006.

64. Y. Yanagisawa, J. Akahani, and T. Satoh. Shape-based similarity query for trajectory of mobile objects. *Proc. Int'l Conf. on Mobile Data Management (MDM'03)*, LNCS 2574, pp. 63–77, 2003.

65. Y. Yanagisawa and T. Satoh. Clustering multidimensional trajectories based on shape and velocity. *Proc. IEEE Int'l Workshop on Multimedia Databases and Data Management (MDDM'06)*, 2006.

66. J. Yang and M. Hu. TrajPattern: Mining sequential patterns from imprecise trajectories of mobile objects. *Proc. Int'l Conf. on Extending Database Technology (EDBT'06)*, LNCS 3896, pp. 664–681, 2006.

67. G. Yavaş, D. Katsaros, Ö. Ulusoy, and Y. Manolopoulos. A data mining approach for location prediction in mobile environments. *Data & Knowledge Engineering*, 54(2):121–146, 2005.

68. M.L. Yiu and N. Mamoulis. Clustering objects on a spatial network. *Proc. ACM SIGMOD Int'l Conf. on Management of Data (SIGMOD'04)*, pp. 443–454, 2004.

69. T. Zhang, R. Ramakrishnan, and M. Livny. BIRCH: A new data clustering algorithm and its applications. *Data Mining and Knowledge Discovery*, 1(2):141–182, 1997.

70. J. Zhang. Location management in cellular networks. In I. Stojmenovic (ed.), *Handbook of Wireless Networks and Mobile Computing*, John Wiley & Sons, pp. 27–49, 2002.

71. Q. Zhang and X. Lin. Clustering moving objects for spatio-temporal selectivity estimation. *Proc. Australasian Database Conf. (ADC'04)*. pp. 123–130, 2004.

Index

Apriori
  algorithm, ix
  property, vii
Association rules, viii
  spatio-temporal, viii
BIRCH, xi
Clustering, xi
  density-based, ix, xiii
  medoid, xiii
Convergence, x
Data cube, xvi
Data mining, i
Dense regions, ix, xiv
Density queries, xvii
Duality transform, xv
Dynamic time warping (DTW), xviii
Edit distance on real sequence (EDR), xix
Encounter, x
Euclidean distance, xviii
FP-growth, ix
Flock, x
Group patterns, ix
High traffic region, ix
Histograms, xv
  density, xv
  mobility, xvi
  spatio-temporal, xv
LZ-based predictors, iii
LZ78, iii
LeZi-Update, iii
Leadership, x
Least common subsequence (LCSS) distance, xix
Markov chains, ii
Markov predictors, ii
Markov properties, ii
Micro-clusters, xi
  moving, xi
Mobility patterns, x
Mobility prediction, ii
Mobility rules, vii
Moving clusters, xii
Moving object databases, i
OLAP (On-Line Analytical Processing), v
Periodic movement patterns, ix
Periodic pattern mining, ix
REMO (RElative MOtion) framework, x
Sequential pattern mining, vi
Sequential pattern, vi
Sketches, xvi
Spatial density function, xii
Spatial network, xiii
Spatio-temporal databases, xx
Stationary region, ix
TrajPattern, vi
Z-ordering, iv