IJARCCE

ISSN (Online) 2278-1021
ISSN (Print) 2319-5940

International Journal of Advanced Research in Computer and Communication Engineering
ISO 3297:2007 Certified
Vol. 6, Issue 12, December 2017

Analysis the Space Partitioning and Group Clustering

Bhupeshwar Kumar Sahu[1], **Dr. Vishnu Mishra**[2]

Ph.D (Research Scholar), Department of IT & CS, Dr. C.V. Raman University, Bilaspur, India [1]
Associate Professor, CSE, Bharti College of Engineering & Technology, Durg, India [2]

Abstract: This paper study of clustering algorithm for trajectories elements which is based on dense of the grouping element. Our experiments give new method for partitioning the *TRACLUS* algorithm and provide the Euclidean distance of moving element. We provide a new approach of moving elements. This approach develops a cluster of trajectory object and calculates the actual distance of moving object. This algorithm works on the *CLSTR* algorithm and calculates the actual cell value. This paper assumes the entropy of moving object and heuristic parameter.

Keywords: Trajectory algorithm, Partition and group work, Data mining, Cluster, Trajectory clustering.

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 overtime 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 every day. 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 problem finding dense regions in Figure 1. Data space is divided into nine cells from A to *I* and four objects (o1, o2, o3, and, o4) have moved in the space for two timestamps. If we define a dense region as a cell having more than two hitting points, r1 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, r2 and r3 are distinct dense regions; however, both should be expressed by only one cell E due to the rough granularity.

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 are 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.

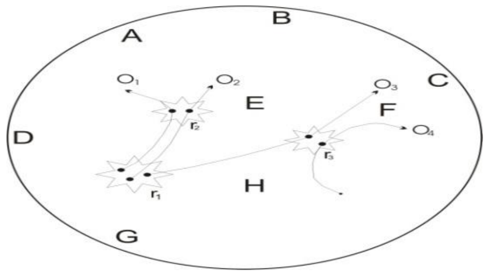

Figure 1: Deficiencies of space-partitioning approaches

II. RELATED WORK

Clustering has been extensively studied in the data mining area. Clustering algorithms can be classified into four categories: partitioning methods (e.g., *k*-means [17]), hierarchical methods (e.g., BIRCH [24]), density-based methods (e.g., DBSCAN [6] and OPTICS [2]), and grid-based methods (e.g., STING [22]). Our algorithm TRACLUS falls in the category of density-based methods. In density-based methods, clusters are regions of high density separated by regions of low density. DBSCAN has been regarded as the most representative density-based clustering algorithm, and OPTICS has been devised to reduce the burden of determining parameter values in DBSCAN. The majority of previous research has been focused on clustering of point data. The most similar work to ours is the trajectory clustering algorithm proposed by Gaffney *et al*.[7, 8]. It is based on probabilistic modeling of a set of trajectories. Formally, the probability density function of observed trajectories is a mixture density: $P(y_j|x_j,\theta) = \sum_k^k f_k(y_j|x_j,\theta_k)w_k$, where$f_k(y_j|x_j,\theta_k)$ is the density component, w_k is the weight, and θ_k is the set of parameters for *k-th* component. Here, θ_k and w_k can be estimated from the trajectory data using the Expectation-Maximization (EM) algorithm. The estimated density components $f_k(y_j|x_j,\theta_k)$ are then interpreted as clusters. The fundamental difference of this algorithm from TRACLUS is being based on probabilistic clustering and clustering trajectories as a whole.

Distance measures for searching similar trajectories have been proposed recently. Vlachos et al. [21] have proposed the distance measure LCSS, and Chen et al. [5] the distance measure EDR. Both LCSS and EDR are based on the edit distance and are extended so as to be robust to noises, shifts, and different lengths that occur due to sensor failures, errors in detection techniques, and different sampling rates. EDR can represent the gap between two similar subsequence’s more precisely compared with LCSS [5]. Besides, dynamic time warping has been widely adopted as a distance measure for time series [12]. These distance measures, however, are not adequate for our problem since they are originally designed to compare the whole trajectory (especially, the whole time-series sequence). In other words, the distance could be large although some portions of trajectories are very similar. Hence, it is hard to detect only similar portions of trajectories. The MDL principle has been successfully used for diverse applications, such as graph partitioning [3] and distance function design for strings [13]. For graph partitioning, a graph is represented as a binary matrix, and then, the matrix is divided into disjoint row and column groups such that the rectangular intersections of groups are homogeneous. Here, the MDL principle is used to automatically select the number of row and column groups [3]. For distance function design, data compression is used to measure the similarity between two strings. This idea is tightly connected with the MDL principle [13].

III. PROPOSED ALGORITHM

A. The *CLSTR* Algorithm

The trajectory clustering algorithm *TRCLS* consist of three phases. Its executes three algorithms to perform the subtasks (lines 2, 4 and 6), in the first phase we execute the trajectory partitioning algorithm and then second phase we execute the trajectory clustering algorithm. We detailed explain these algorithms in Section A and B.

Algorithm *CLSTR* (Clustering Trajectory)

```
--------------------------------------------------
Input:    A set of trajectories $T = \{J_1,\cdots,Jnum_{tra}\}$
Output:  (1) A set of clusters $R = \{S_1,\cdots,Snumn_{clr}\}$
         (2) A set of representative trajectories
```

Algorithm:

```
    /* Partitioning Phase */
```

```
01:  for each ($J \in T$) do
02:      Execute *Approximate Trajectory Partitioning;*
           Gets a set *Ls* of line segments using the result;
03:        Accumulate *Ls* into a set *E*;
     /* Grouping Phase */
04:        Execute *Line Segment Clustering for E*;
           Get a set of *R* clusters as the result;
05:        for each ( $S \in R$ ) do;
06:        Execute *Representative Trajectory Generation;*
           Get a representative trajectory as the result;
```

B. Partitioning Trajectory (Partitioning Phase)

In this section, we propose a *Approximate Trajectory Partitioning* algorithm for trajectory clustering. The algorithm *Approximate Trajectory Partitioning* shows below. We compute MDLpar and MDL_{nopar} for each point in a trajectory (lines 5~6). If MDL_{par} is greater than MDL_{nopar}, we insert the immediately previous point pcurrIndex−1 into the set CP_i of characteristic points (line 8). Then, we repeat the same procedure from that point (line 9). Otherwise, we increase the length of a candidate trajectory partition (line 11)

Algorithm Approximate Trajectory Partitioning

Input: A trajectory TR_i = p1p2p3⋯pj ⋯$plen_i$
Output: A set CP_i of characteristic points

Algorithm:

```
01:    Add p1 into the set $CP_i$; /* the starting point */
02:    startIndex := 1, length := 1;
03:    while (startIndex + length ≤ leni) do
04:        currIndex := startIndex + length;
05:        $cost_{par}$ := $MDL_{par}$(pstartIndex, pcurrIndex);
06:        $cost_{nopar}$ := $MDL_{nopar}$(pstartIndex, pcurrIndex);
           /* check if partitioning at the current point makes the MDL cost larger than not partitioning */
07:        if ($cost_{par}$> $cost_{nopar}$) then
                   /* partition at the previous point */
08:            Add pcurrIndex − 1 into the set $CP_i$;
09:            startIndex := currIndex−1, length := 1;
10:        else
11:        ength := length + 1;
12:    Add $plen_i$ into the set $CP_i$;   /* the ending point */
```

C. Clustering Trajectory (Grouping Phase)

In this section, we propose a *line segment clustering algorithm* for trajectory clustering and *Representative Trajectory Generation* algorithm in the grouping phase. We now present our density-based clustering algorithm for line segments. Given a set D of line segments, our algorithm generates a set O of clusters. It requires two parameters ε and *MLins*. We define a cluster as a density-connected set. Our algorithm shares many characteristics with the algorithm DBSCAN.

Unlike DBSCAN, however, not all density-connected sets can become clusters. We need to consider the number of trajectories from which line segments have been extracted. This number of trajectories is typically smaller than that of line segments. For example, in the extreme, all the line segments in a density-connected set could be those extracted from one trajectory. We prevent such clusters since they do not explain the behavior of a sufficient number of trajectories.

Algorithm: Line Segment Clustering

Input: (1) A set of line segments $D = \{L_1, \cdots, L_{num_{ln}}\}$,
(2) Two parameters ε and *MLins*

Output: A set of clusters $S = \{C_1, \cdots, C_{num_{clus}}\}$

Algorithm:

/* 1 Step */

```
01: Set $cluster\ Id$ to be 0; /* an initial id */
02: Mark all the line segments in $D$ as a $unclassified$;
03: for each ($L \in D$) do
04:     if ($L$ is $unclassified$) then
05:         Compute $N_\varepsilon(L)$;
06:         if ($|N_\varepsilon(L)| \geq MLins$) then
07:             Assign $cluster\ Id$ to $\forall X \in N_\varepsilon(L)$;
08:             Insert $N_\varepsilon(L) - \{L\}$ into the queue $Q$;
/* 2 Step */
09:             ExpandCluster($Q$, $cluster\ Id$, $\varepsilon$, $MLins$);
10:             Increase $cluster\ Id$ by 1; /* a new id */
11:         else
12:             Mark $L$ as $noise$;
/* 3 Step */
13: Allocate $\forall L \in D$ to its cluster $C_{cluster\ Id}$;
/* check the trajectory cardinality */
14: for each ($C \in S$) do
        /* a threshold other than $MLins$ can be used */
15:     if($|PTR(C)| < MLins$) then
16:         Remove $C$ fromthe set $S$ of clusters;
/* 4 Step: compute a density – connected set */
17: ExpandCluster($Q$, $clusterId$, $\varepsilon$, $MLins$) {
18:     while ($Q = \emptyset$) do
19:         Let $M$ be the first line segment in $Q$;
20:         Compute $N_\varepsilon(M)$;
21:         if ($|N_\varepsilon(M)| \geq MLins$) then
22:             for each ($X \in N_\varepsilon(M)$) do
23:                 if ($X$ is $unclassified$ or $noise$) then
24:                     Assign $cluster\ Id$ to $X$;
25:                 if ($X$ is $unclassified$) then
26:                     Insert $X$ into the queue $Q$;
27:         Remove $M$ from the queue $Q$;
28: }
```

Algorithm: Representative Trajectory Generation

--

Input: (1) A cluster C_i of line segments,

(2) MLins (3) A smoothing parameter α

Output: A representative trajectory RTR_i for C_i

Algorithm:

```
01: Compute the average direction vector $\vec{v}$;
02: Rotate the axes so that the $X$ axis is parallel to $\vec{v}$;
03: Let $P$ be the set of the starting and ending points of the line segments in $C_i$;
        /* $X'$-value denotes the coordinate of the $X'$ axis */
04: Sort the points in the set P by their $X'$-values;
05:     for each ($p \in P$) do
            /* count $num_p$ using a sweep line (or plane) */
06:         Let $num_p$ be the number of the line segments that contain the $X'$- value of the point $p$;
07:             if ($num_p \geq MLins$) then
```

IJARCCE

ISSN (Online) 2278-1021
ISSN (Print) 2319-5940

International Journal of Advanced Research in Computer and Communication Engineering
ISO 3297:2007 Certified
Vol. 6, Issue 12, December 2017

```
08:        diff := the difference in X´- values between p and its immediately previous point;
09:        if (diff ≥ α) then
10:            Compute the average coordinate avg´_p;
11:            Undo the rotation and get the point avg_p;
12:            Append avg_p to the end of RTR_i;
```

D. Experimental Evaluation

Results & Analysis of Saffir-Simpson Hurricane Track Data-

Figure 4 shows the entropy as ε is varied. The minimum is achieved at ε = 43. Here, *avg|Nε(L)|* is 7.26. According to our heuristic, we try parameter values around ε = 43 and *MLins*= 10~12. Using visual inspection and domain knowledge, we are able to obtain the optimal parameter values: ε = 42 and *MLins* = 11. We note that the optimal value ε = 42 is very close to the estimated value ε = 43. Figure 2 shows the quality measure as ε and *MLins* are varied. The smaller *KMeasure* is, the better the clustering quality is. There is a little discrepancy between the actual clustering quality and our measure. Visual inspection results show that the best clustering is achieved at *MLins* = 11, not at *MLins* = 10. Nevertheless, our measure is shown to be a good indicator of the actual clustering quality within the same *MLins* value. That is, if we consider only the result for *MLins* = 11, we can see that our measure becomes nearly minimal when the optimal value of ε is used. We know that some hurricanes move along a curve, changing their direction from east-to-west to south-to-north, and then to west to-east. On the other hand, some hurricanes move along a straight east-to-west line or a straight west-to-east line. The lower horizontal cluster represents the east-to-west movements, the upper horizontal one the west-to-east movements, and the vertical ones the south-to-north movements.

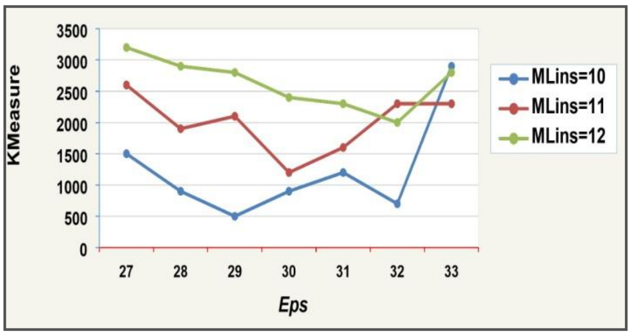

Figure 2: Quality measure for the hurricane data

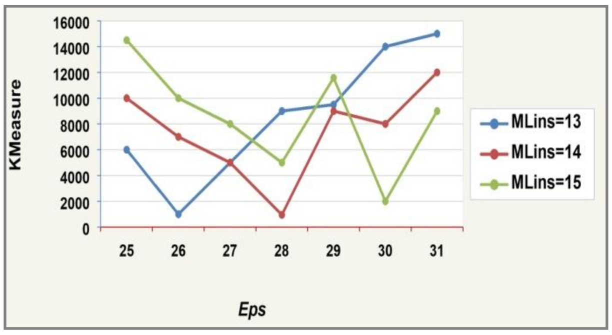

Figure 3: Quality measure for the Blue Bull 2002 data

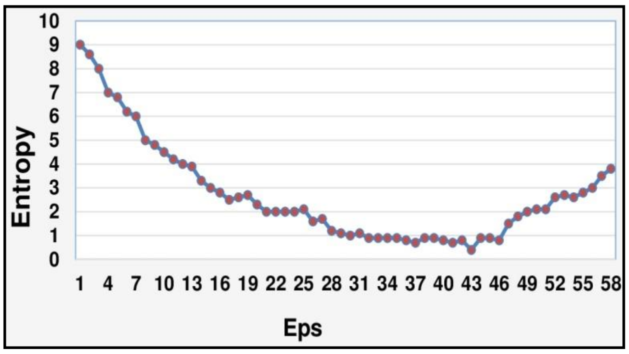

Figure 4: Entropy for hurricane data

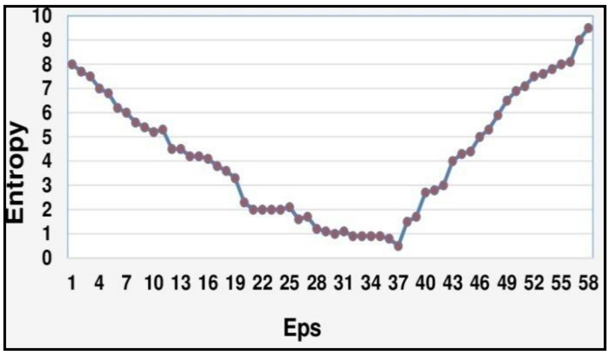

Figure 5: Entropy for Blue Bull 2002 data

Results & Analysis of Animal Movement Data-

Blue Bull's Movements in 2002:
Figure 5 shows the entropy as ε is varied. The minimum is achieved at ε = 37. Here, *avg|Nε(L)|* is 9.52. Visually, we obtain the optimal parameter values: ε = 39 and *MLins* = 13. Again, the optimal value ε = 37 is very close to the estimated value ε = 37.

Figure 20 shows the quality measure as ε and *MLins* are varied. We observe that our measure becomes nearly minimal when the optimal parameter values are used. The correlation between the actual clustering quality and our measure, *KMeasure*, is shown to be stronger in Figure 3 than in Figure 2.

Black Buck's Movements in 2002:

IJARCCE

ISSN (Online) 2278-1021
ISSN (Print) 2319-5940

International Journal of Advanced Research in Computer and Communication Engineering
ISO 3297:2007 Certified
Vol. 6, Issue 12, December 2017

Our experiment shows the clustering result using the optimal parameter values (ε = 41 and *MLins* = 13). The result indicates that two clusters are discovered in the two densest regions. This result is exactly what we expect. The centre region is not so dense to be identifies as a cluster. Due to space limit, we omit the detailed figures for the entropy and quality measure.

V. CONCLUSION

In this paper, we have proposed a novel framework, the partition-and-group framework, for clustering trajectories. Based on this framework, we have developed the trajectory clustering algorithm *CLSTR*. As the algorithm progresses, a trajectory is partitioned into a set of line segments at characteristic points, and then, similar line segments in a dense region are grouped into a cluster. The main advantage of *CLSTR* is the discovery of common sub-trajectories from a trajectory database. To show the effectiveness of *CLSTR*, we have performed extensive experiments using two real data sets: Saffir-Simpson hurricane track data and animal movement's data. Our heuristic for parameter value selection has been shown to estimate the optimal parameter values quite accurately. We have implemented a visual inspection tool for cluster validation. The visual inspection results have demonstrated that *CLSTR* effectively identifies common sub-trajectories as clusters. Overall, we believe that we have provided a new paradigm in trajectory clustering. Data analysts are able to get a new insight into trajectory data by virtue of the common sub trajectories. This work is just the first step, and there are many challenging issues discussed above. We are currently investigating into detailed issues as a further study.

REFERENCES

[1] Lee, T. C. M., "An Introduction to Coding Theory and the Two-Part Minimum Description Length Principle," International Statistical Review, 69(2): 169–184, 2001.
[2] Powell, M. D. and Aberson, S. D., "Accuracy of United States Tropical Cyclone Landfall Forecasts in the Atlantic Basin (1976-2000)," Bull. of the American Meteorological Society, 82(12): 2749–2767, 2001.
[3] Vlachos, M., Gunopulos, D., and Kollios, G., "Discovering Similar Multidimensional Trajectories," In Proc. 18th Int'l Conf. on Data Engineering, San Jose, California, pp. 673–684, Feb./Mar. 2002.
[4] Wisdom, M. J., Cimon, N. J., Johnson, B. K., Garton, E. O., and Thomas, J. W., "Spatial Partitioning by Mule Deer and Elk in Relation to Traffic," Trans. of the North American Wildlife and Natural Resources Conf., Spokane, Washington, pp. 509–530, Mar. 2004.
[5] Keogh, E. J., Lonardi, S., and Ratanamahatana, C. A., "Towards Parameter-Free Data Mining," In Proc. 10th ACM SIGKDD Int'l Conf. on Knowledge Discovery and Data Mining, Seattle, Washington, pp. 206–215, Aug. 2004.
[6] Guttman, A., "R-Trees: A Dynamic Index Structure for Spatial Searching," In Proc. 1984 ACM SIGMOD Int'l Conf. on Management of Data, Boston, Massachusetts, pp. 47–57, June 1984.
[7] Gaffney, S., Robertson, A., Smyth, P., Camargo, S., and Ghil, M., Probabilistic Clustering of Extratropical Cyclones Using Regression Mixture Models, Technical Report UCI-ICS 06-02, University of California, Irvine, Jan. 2006.
[8] Ester, M., Kriegel, H.-P., Sander, J., and Xu, X., "A Density-Based Algorithm for Discovering Clusters in Large Spatial Databases with Noise," In Proc. 2nd Int'l Conf. on Knowledge Discovery and Data Mining, Portland, Oregon, pp. 226–231, Aug. 1996.
[9] Chakrabarti, D., Papadimitriou, S., Modha, D. S., and Faloutsos, C., "Fully Automatic Cross-Associations," In Proc. 10th ACM SIGKDD Int'l Conf. on Knowledge Discovery and Data Mining, Seattle, Washington, pp. 79–88, Aug. 2004.
[10] Achtert, E., B¨ohm, C., Kriegel, H.-P., Kr¨oger, P., and Zimek, A., "Deriving Quantitative Models for Correlation Clusters," In Proc. 12th ACM SIGKDD Int'l Conf. on Knowledge Discovery and Data Mining, Philadelphia, Pennsylvania, pp. 4–13, Aug. 2006.
[11] Chen, J., Leung, M. K. H., and Gao, Y., "Noisy Logo Recognition Using Line Segment Hausdorff Distance," Pattern Recognition, 36(4): 943–955, 2003.
[12] Gaffney, S. and Smyth, P., "Trajectory Clustering with Mixtures of Regression Models," In Proc. 5th ACM SIGKDD Int'l Conf. on Knowledge Discovery and Data Mining, San Diego, California, pp. 63–72, Aug. 1999.
[13] Gru¨nwald, P., Myung, I. J., and Pitt, M., Advances in Minimum Description Length: Theory and Applications, MIT Press, 2005.
[14] Han, J. and Kamber, M., Data Mining: Concepts and Techniques, 2nd ed., Morgan Kaufmann, 2006.
[15] Kirkpatrick, S., Gelatt, C. D., and Vecchi, M. P., "Optimization by Simulated Annealing," Science, 220(4598): 671–680, 1983.
[16] Lee, J.-G., Han, J., and Whang, K.-Y., Trajectory Clustering: A Partition-and-Group Framework, Technical Report UIUCDCS-R-2007-2828, University of Illinois at Urbana-Champaign, Mar. 2007.
[17] Roth, V., Laub, J., Kawanabe, M., and Buhmann, J. M., "Optimal Cluster Preserving Embedding of Nonmetric Proximity Data," IEEE Trans. on Pattern Analysis and Machine Intelligence, 25(12): 1540–1551, 2003.
[18] Wang, W., Yang, J., and Muntz, R. R., "STING: A Statistical Information Grid Approach to Spatial Data Mining," In Proc. 23rd Int'l Conf. on Very Large Data Bases, Athens, Greece, pp. 186–195, Aug. 1997.
[19] Zhang, T., Ramakrishnan, R., and Livny, M., "BIRCH: An Efficient Data Clustering Method for Very Large Databases," In Proc. 1996 ACM SIGMOD Int'l Conf. on Management of Data, Montreal, Canada, pp. 103–114, June 1996.
[20] Shannon, C. E., "A Mathematical Theory of Communication," The Bell System Technical Journal, 27: 379–423 and 623–656, 1948.
[21] Lloyd, S., "Least Squares Quantization in PCM," IEEE Trans. on Information Theory, 28(2): 129–137, 1982.
[22] Keogh, E. J., "Exact Indexing of Dynamic Time Warping," In Proc. 28th Int'l Conf. on Very Large Data Bases, Hong Kong, China, pp. 406–417, Aug. 2002.
[23] Gaffney, S., Robertson, A., Smyth, P., Camargo, S., and Ghil, M., Probabilistic Clustering of Extratropical Cyclones Using Regression Mixture Models, Technical Report UCI-ICS 06-02, University of California, Irvine, Jan. 2006.
[24] Chen, L., ¨Ozsu, M. T., and Oria, V., "Robust and Fast Similarity Search for Moving Object Trajectories," In Proc. 2005 ACM SIGMOD Int'l Conf. on Management of Data, Baltimore, Maryland, pp. 491–502, June 2005.
[25] Ankerst, M., Breunig, M. M., Kriegel, H.-P., and Sander, J., "OPTICS: Ordering Points to Identify the Clustering Structure," In Proc. 1999 ACM SIGMOD Int'l Conf. on Management of Data, Philadelphia, Pennsylvania, pp. 49–60, June 1999.