Mining Group Relationship in an Object Tracking Sensor Network

Hsiao-Ping Tsai, De-Nian Yang, and Ming-Syan Chen
National Taiwan University, Taipei, Taiwan, ROC
{hptsai@arbor, dnyang@cc, mschen@cc}.ee.ntu.edu.tw

***Abstract*—In the object tracking applications, previous works show that moving objects usually exhibit some degrees of regularity in their movement. Biologists find out that many creatures have social behavior that is they usually form mass organization and migrate together. In this paper, we investigate and utilize the characteristic of the group movement of objects to explore the group relationship in the wireless object tracking sensor network (OTSN). In contrast to the centralized mining, we propose a distributed mining algorithm to find a group of moving objects with similar moving patterns. Our algorithm consists of the local mining phase and the cluster ensembling phase. The local mining phase adopts the VMM model together with Probabilistic Suffix Tree (PST) to find the moving patterns, as well as Highly Connected Component (HCS) to partition the moving objects. The cluster ensembling phase utilizes Jaccard Similarity Coefficient and Normalized Mutual Information to combine and improve the local grouping results. The experiment results show that our distributed mining algorithm achieves good grouping quality and robustness.**

I. Introduction

In the object tracking applications, previous works show that moving objects usually exhibit some degrees of regularity in their movements. For example, the famous annual wildebeest migration points out that the movement of creatures is temporal-and-spatial correlated. Research works [1][2][3] model the correlation as the sequential patterns in data mining, and various algorithms have been proposed to find the frequent moving patterns. However, previous works only find the moving patterns for each individual object. Algorithms to find the moving patterns of a group of objects and related design issues have not been discussed in the related works. Biologists find out that many creatures such as animals, birds, or insects have social behavior that is they usually form mass organization and migrate for food, breeding, wintering, or other unknown reasons. Our point is if we can find the group relationship as well as the object moving patterns, we can create many opportunities to track objects efficiently. For example, we can utilize the group relationship in the in-network sensor scheduling or data aggregation for tracking multiple objects simultaneously and efficiently. Therefore, our work focuses on grouping objects based on their movements in the OTSN.

In this paper, we propose a new algorithm to identify a group of moving objects and find the corresponding moving patterns. Our algorithm consists of a local Group Moving Pattern Mining (GMPMine) algorithm to extract group relationship locally and a Cluster Ensembling algorithm to combine and improve the local grouping results. GMPMine adopts Variable length Markov Model (VMM) for modeling the statistics of objects' movements and Probabilistic Suffix Tree (PST) [4] for learning the significant moving patterns, as well as Highly Connected Component (HCS) to partition the moving objects. Please note that VMM and PST are widely used in many fields such as gene clustering [5], musical style learning [6], outlier detection [7], application behavior modeling [8], and so on. PST is excellent in the capacity of extracting structural information from the sequences. We adopt PST for the resource-limited wireless sensor networks due to its low complexity in time and space [9]. In addition, it is very efficient in prediction and robust to outliers.

In contrast to the central mining scenario that the cluster heads transmit all features to the sink, we only transmit the mined out grouping results to the sink for conserving power. In the cluster ensembling, we utilize Jaccard Similarity Coefficient to measure the similarity between a pair of objects as well as Normalized Mutual Information (NMI) to achieve the best ensembling result. Cluster ensembling is a research topic whose goal is seeking a combination of multiple clustering results of the given data to achieve better clustering quality, robustness, stability, and scalability [10][11][12]. However, the above works focus on balanced clustering for a predefined cluster number that differentiates their problems from ours. Jaccard Similarity Coefficient is a statistic that is commonly used in information retrieval for comparing the similarity between two binary vectors [13][14]. In the tracking applications, moving objects may present in partial clusters and absent from the others. The importance of positive and negative opinions is asymmetric and counting the absence for both objects has no meaningful contribution to the similarity measurement. Thus, Jaccard is more objective and reasonable than other measures such as Simple Matching Coefficient and Overlap coefficient. After that, with a set of configured thresholds, we utilize HCS to partition the objects and leverage NMI to optimize the ensembling result. Note that NMI is an entropy-based measure between two distributions and is broadly used in information retrieval, e.g. natural language processing [15], cluster ensembling [10], image registration [16], and so on.

The rest of paper is outlined as follows. In Section 2, we formulate our problem and present the GMPMine algorithm. In Section 3, we describe the Cluster Ensembling algorithm. We show our experimental results in Section 4. Finally, Section 5 concludes this paper.

II. Problem Formulation and the GMPMine Algorithm

In a hierarchical WSN, nodes are heterogeneous in energy, computing, and storage capacity. Higher-energy node can be used to perform high complexity computing and send data, while low-energy node can be used to perform the sensing and low complexity computing. A sensor equipped with higher energy assigned as a cluster head (CH) to handle high complexity tasks. Every sensor within a cluster has a locally unique ID, and the CH logically represents the sensors within the identical cluster and acts as a sensor in the view of the upper layer. Generally, when a sensor detects an object of interest, it informs the location data to the CH. The CH then forwards the location data upward until the sink. In this work, we adopt the hierarchical cluster structure of two layers, and each cluster is a mesh network and contains 4×4 sensors.

A pair of object ID and the ID of the nearest sensor model the location data of an object. We represent the sensor IDs by an alphabet Σ. The number of symbols of the alphabet Σ is dependent on the cluster size. For example, if the cluster size is 16, we use an alphabet $\Sigma = \{a, b, ..., p\}$ to represent the IDs for a cluster of sensors. Therefore, the location sequence of a moving object is represented by a sequence of symbols over Σ, denoted by $S = s_0 s_1 ... s_{L-1}$, where $s_i \in \Sigma$ for $0 \leq i < L$.

Our distributed algorithm adaptively forms a Sensor Group (SG) to trace a group of moving objects. The algorithm adaptively adjusts the members of a SG in a distributed manner as the moving objects move. The Group Data Aggregation Range (GDAR) of a SG is the hop count between the central sensor of the SG and the furthest sensors in the same SG, and with which we can control the size of a SG. On detecting an object, the sensor invokes to form a SG to collect location data and forwards the data to the CH. The CH then extracts the sensor ID from the location data and appends it to the location sequence for each sensed objects. In this paper, we set the GDAR of a SG corresponding to the specified error bound and report the central sensor of the SG as the locations of the captured objects. The error bound of the tolerant error is the maximal allowed distance between the real location and the reported one in hop count. For example, assume there are 5 objects walking a sensor cluster as shown in Fig. 1. With GDAR$= 1$, s_g first detects an object and invokes to form a SG with its 1-hop neighbors. After that, s_g aggregates and reports the location data including (o_1, s_k), (o_2, s_k), (o_3, s_g), and (o_4, s_f). The CH appends s_g to the location sequences of o_1~o_4 separately. The location data collection process repeats for a period and finally forms the location sequence data set.

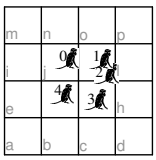


Fig. 1. The example of 5 objects within a cluster.

In this paper, a set of moving objects belong to the same group if they share similar moving patterns, and the longest distance between an object's location and the center of the group is defined as the Group Dispersion Radius (GDR). Therefore, we formulate our problems as follows. *Mining the group relationship among the objects based on their location sequence data set and combining the ensemble of the local grouping results to achieve better grouping quality.* In the following sections, we first describe the VMM model and then the GMPMine algorithm.

A. VMM Prediction Model

In this paper, we leverage the spatial correlation of moving objects that is the next location is predicted according to preceding locations. We adopt VMM for modeling the statistics of objects' movement as follows. This property of an object's movement is modeled by the conditional probability distribution over an alphabet Σ on a given location sequence data set. For example, if s is a location sequence over Σ, σ is a symbol in Σ, $P(\sigma|s)$ is the probability that σ will come following s. While the history length is floating that is k is unfixed, the model is a VMM. The advantage over a fixed k-order Markov model is the ability to flexibly adaptive to memory length for prediction. Ron et al. [4] proposed the Probabilistic Suffix Tree (PST) that is an implementation of VMM. PST is widely used in finding significant patterns for gene clustering [17], musical style learning [6], outlier detection [7], and application behavior modeling [8]. The PST building algorithm learns from a data set and outputs a PST representing a dictionary of significant patterns. A PST over an alphabet Σ is a non-empty tree, in which nodes vary in degree between zero and $|\Sigma|$. Each edge in the tree is labeled by a single symbol of the alphabet such that no symbol can be represented by more than one edge branching out of any single node. Note that each node in the tree is associated with a significant pattern and is labeled by a string s, which is also the sequence generated by walking up the tree from that node to the root. The node also contains the conditional empirical probabilities, i.e. $P(\sigma|s)$, for every $\sigma \in \Sigma$.

Let Xs denote the number of times that a subsequence s is observed in the data set and $N|s|$ denote the total number of overlapping subsequences of length $|s|$ within the data set. The empirical probability $P(s)$ of a subsequence s over the given data set is $\frac{Xs}{N|s|}$. The conditional empirical probability $P(\sigma|s)$ of observing a symbol σ right after a given subsequence s is the number of times that σ has shown up right after s, denoted by $Xs\sigma$, divided by the total number of times that s has shown up at all, followed by any symbol $Xs*$. That is $P(\sigma|s) = \frac{Xs\sigma}{Xs*}$. For example, given a data set $\{bcgkjnm$, $cgkonm$, $cgfjknm$, $cgkonm\}$ and a subsequence $s =$ "cgk", the conditional empirical probability $P('o'|s)$ is $0.\overline{6}$ and the probability $P(\text{"}cgko\text{"})$ is 0.0769.

$$Xcgk = 3\ N_3 = (7-2)+(6-2)+(7-2)+(6-2) = 18$$
$$P(cgk) = 3/18 = 0.1\bar{6}$$
$$Xcgko = 2$$
$$Xs *= 3$$
$$P(o|cgk) = 2/3 = 0.\bar{6}$$

$$P(cgko) = p(c) \times p(g|c) \times p(k|cg) \times p(o|cgk)$$
$$= \frac{4}{26} \times \frac{4}{4} \times \frac{3}{4} \times \frac{2}{3}$$
$$\div 0.0769$$

One advantage of PST is the low complexity, i.e. $O(n)$, in both time and space, where n is the sequence length. The compact tree and its low computing complexity make PST suitable to the resource-constrained environment. The details of the PST building algorithm and other related discussions can be referred to [4][5][18].

B. The Group Moving Pattern Mining (GMPMine) Algorithm

For mining the group relationship, we propose the GMPMine algorithm shown in Fig. 2 to group objects based on the similarity of their moving patterns. Our algorithm first builds a PST for each object and then leverages the PSTs in grouping objects. We define the similarity score sim_p of two objects as follows,

$$sim_p = -log\frac{\sum_{s\in\widetilde{S}}\sqrt{\sum_{\sigma\in\Sigma}(P_1(s)\times P_1(\sigma|s)-P_2(s)\times P_2(\sigma|s))^2}}{2L_{max}+\sqrt{2}}$$

, where $\widetilde{S}$ denotes the union of significant patterns on two tree, and L_{max} is the maximal memory length of PST. sim_p is the combination of condition probability and Euclidean distance of each significant pattern $s \in \widetilde{S}$. With the definition of sim_p, we compute the similarity score for each pair of objects and construct an unweighted undirected similarity graph G. A node in G corresponds to an object. An edge between two objects represents that sim_p of the two objects is higher than the threshold sim_{min}. In addition, we leverage the time-overlapping pruning technique and the noise pruning technique to assist in evaluating the group relationship. The lower time-overlapping ratio between two objects implies the lower probability that the two objects belong to a group. The noise pruning technique utilizes the similarity score between a PST and the PST of certain patterns such as random walk to filter objects. Then we leverages the HCS algorithm together with a min-cut algorithm to partition the similarity graph into highly connected subgraphs, each of which is corresponding to a group. We thereby extract the group relationship among objects. For the details of the algorithm, interested readers can refer to [19].

III. The Cluster Ensembling Algorithm

In this section, we propose the Cluster Ensembling algorithm to combine multiple local grouping results from the CHs to improve the grouping quality. Our algorithm considers the case that the trajectories of moving objects span multiple sensor clusters and utilizes the Jaccard similarity coefficient to take only relative CHs' opinions in measuring the similarity between pairs of objects. Jaccard Similarity Coefficient is a statistic that is commonly used in information retrieval

```
Algorithm: GMPMine
Input:  Ŝ ={ Si|0≤i<N}, simmin, Ttarget, simnoise, tomin
Output: G, m
0.   G = Ø
1.   m = 0
2.   /*building a PST for each object and pruning noise*/
3.   for each Si in Ŝ
4.       Ti = PST_Build(Si)
0.       if simp(Ti, Ttarget)>simnoise then
5.           delete Ti
6.   /*constructing a similarity graph on PSTs*/
7.   for 0≤i< N-1
8.       for i+1≤j< N
9.           if simp(Ti, Tj) >simmin and overlap_time(i,j) > tomin then
10.              add_edge(i,j) to Graph(V, E )
11.  /*extracting highly connected subgraph*/
12.  (G,m) = HCS(Graph(V, E ))      // G={gi|0≤i<m}
13.  /* selecting a group PST GTi for each group gi  */
14.  for 0≤i<m
15.      S'= {Sj| oj ∈ gi, 0≤j<N}
16.      T'= {Tj| oj ∈ gi, 0≤j<N}
17.      GTi = arg max_{Tj} Σ_{Sk∈S'} Pj(Sk) where Tj ∈ T'
18.  return G = {gi|0≤i<m}, GT = {GTi|0≤i<m}
```

Fig. 2. The GMPMine Algorithm.

for comparing the similarity between two binary vectors [20][13][14]. Jaccard is especially suitable for the applications in which the importance of negative and possible values of a feature is asymmetric, and considering the negative value in both objects has no meaningful contribution to the similarity measurement. After computing the similarity between each pair of objects, we partition the objects and leverage Normal Mutual Information (NMI) to optimize the ensembling result. Note that NMI is an entropy-based measurement criterion between two distributions and is broadly used in the field of information theory. A low NMI value indicates two distributions have only a random association, whereas a higher value indicates they are mutually informative.

Let C denote the ensemble of the local grouping results, represented as $C = \{G_1, G_2, ..., G_K\}$, where K denotes the ensemble size, i.e. the total number of CHs. The grouping result G_i obtained from CH_i is a partition of O into m_i disjoint groups, represented as $G_i = \{g_1^i, g_2^i, ..., g_{m_i}^i\}$. As Strehl et al. [10] and Fred et al.[21], the NMI of G_i and G_j, denoted by $NMI(G_i, G_j)$, is the measurement the amount of information shared between G_i and G_j as follows,

$$NMI(G_i,G_j) = \frac{\sum_{a=1}^{m_i}\sum_{b=1}^{m_j}\widehat{P}(a,b)\log\frac{\widehat{P}(a,b)}{\widehat{P}(a)\times\widehat{P}(a)}}{\sqrt{H(G_i)\times H(G_j)}}$$

, where $\widehat{P}(a,b) = \frac{|g_a^i \cap g_b^j|}{|O|}$, and $H(G_i)$ is the entropy of G_i. Note that $H(G_i)$ is the measure of the amount of information of G_i and defined by $H(G_i) = \sum_{a=1}^{m_i} -\widehat{P}(a)log\widehat{P}(a)$, where $\widehat{P}(a) = \frac{|g_a^i|}{|O|}$. To combine the local grouping results C, the objective function of our ensembling algorithm is to find the ensembling result G' that keeps most information of C as follows,

$$G' = arg\max_{G}\sum_{i=1}^{K} NMI(G_i, G).$$

However, enumerating all possible G to find the ensembling result G' is impractical. Therefore, we propose the Cluster

Ensembling algorithm as shown in Fig. 3 to leverage C in partitioning O and generate the ensembling result $G_{\delta'} = \{g_1, g_2, ..., g_{m_{\delta'}}\}$. In the algorithm, we trades off the grouping quality and the computing cost by the configuration of D, where D is a set of thresholds with value$\in [0,1]$ and fine-grained configuration of D achieves better grouping quality and induces more computing cost. The Cluster Ensembling algorithm includes three steps. First, we utilize Jaccard Similarity Coefficient to measure the similarity for each pair of objects as well as construct a similarity matrix according to the local grouping results. Our algorithm then generates the HCS partitioning results based on the similarity matrix for all $\delta \in D$. Afterward, we select the ensembling result based on NMI. Next, we describe the three steps in detail.

Algorithm: Cluster Ensembling
Input: $O = \{o_0, o_1, ... o_N\}$, $C = \{G_i \mid 0 \le i < k\}$, $D = \{\delta_i \mid 0 \le i < d\}$
Output : $G_{\delta'}$
0. init *sum*[]
1. init *SM*[][]
2. *idx* = 0
3. *max* = 0
4. /*building similarity matrix by Jaccard*/
5. **for** $0 \le i < N-1$
6. **for** $i+1 \le j < N$
7. *SM*[i,j] = $getS_{ij}(C)$
8. /*select the partition with max $\sum NMI(G_\delta, G_i)$*/
9. **for** $0 \le i < d$
10. *Graph*(*V, E*) = Convert2Graph(*SM*,δ_i)
11. G_{δ_i} = HCS(*Graph*(*V, E*))
12. *sum*[i] = $\sum_{0 \le j < k} NMI(G_j, G_{\delta_i})$
13. **if** *sum*[i] > *max* **then**
14. *max* = *sum*[i]
15. *idx* = *i*
16. $G_{\delta'} = G_{\delta_{idx}}$
17. **return** $G_{\delta'}$

Fig. 3. The Cluster Ensembling Algorithm.

In the first step, we measure the similarity for each pair of objects to construct a similarity matrix according to the local grouping results (Lines 5-7). In our application, trajectories of moving objects span several clusters such that we take the relative CHs' opinions in considering the group relationship. We thereby utilize the Jaccard similarity coefficient [20] to measure the similarity between two objects. Let $\pi_k(o_i)$ denote the mapping of the local group ID and o_i obtained from CH_k, and $I_k(o_i) = 1$ indicate $\pi_k(o_i) \neq -1$. The group relationship of o_i and o_j in G_k, denoted by $c_k(o_i, o_j)$, represents if objects o_i and o_j belong to the same group in G_k as follows,

$$c_k(o_i, o_j) = \begin{cases} 1 & \text{if } \pi_k(o_i) = \pi_k(o_j) \neq -1; \\ 0 & \textit{else}. \end{cases}$$

We define the Jaccard similarity coefficient of o_i and o_j to be the proportion of local grouping results with o_i and o_j belonging to the same group, denoted by S_{ij}, as follows,

$$S_{ij} = \frac{\sum_{k=1}^{K} c_k(o_i, o_j)}{\sum_{k=1}^{K} I_k(o_i) + \sum_{k=1}^{K} I_k(o_j) - \sum_{k=1}^{K} c_k(o_i, o_j)}. \quad (1)$$

Note that the denominator is the number of local grouping results in which o_i or o_j belongs to some group. According to Eqn. (1), we compute S_{ij} for each pair of objects and construct the similarity matrix, denoted by SM, where $SM[i,j] = S_{ij}$ and $S_{ij} \in [0,1]$.

In the second step, we generate a partitioning result G_δ based on SM for each $\delta \in D$. First, we generate an unweighted undirected graph for each δ according to SM by the Convert2Graph() function as follows. For each S_{ij} in SM, if $S_{ij} > \delta$, we add an edge between o_i and o_j to the similarity graph $G(V, E)$ as Line 10. After that, we partition the graph by HCS to generate G_δ (Line 11).

In the last step, we select the ensembling result $G_{\delta'}$ by exploiting NMI. Given a set of thresholds D, we formulate the ensembling result $G_{\delta'}$ for C as follows,

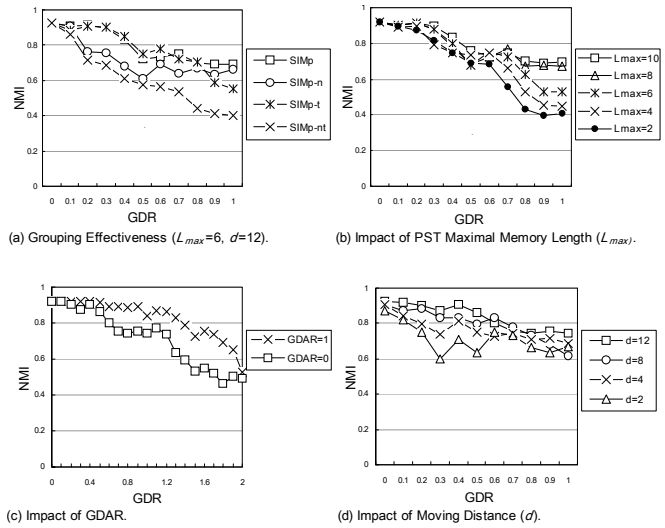
$$G_{\delta'} = arg \max_{\delta \in D} \sum_{i=1}^{K} NMI(G_i, G_\delta).$$

As Lines 13-15, we therefore calculate the NMI sum for each G_δ, i.e. $\sum_{i=1}^{K} NMI(G_i, G_\delta)$ and choose the G_δ with maximal NMI sum as $G_{\delta'}$. Hereafter, we can leverage $G_{\delta'}$ in the tracking phase to achieve energy efficiency.

IV. Experiments

We implemented an event driven simulator in C++ with SIM [22] to evaluate our design. In this simulation, we used the Location-Dependent Parameterization of a Random Direction Mobility Model [23] to simulate the roaming behavior of a group leader and utilized the distance (d) to control the moving range that is the linear distance between the starting point and the end point for the leader. The other members are followers that are uniformly distributed within a specified GDR of the leader. Besides, we inputted objects of random walk in the experiments for making our simulation near to the practical scenarios. In the following experiments, there are 5 groups, each of which contains 5 objects as the grouped, patterned objects and 25 random-walk objects walking in the OTSN. The NMI between the ensembling result and the preknown group relationship of the input workload is used as the evaluation metric. We made the experiments involving the effectiveness of our algorithm and the impacts of PST parameter L_{max}, moving distance (d), and GDAR on the grouping quality to test our approach.

In the first experiment, we studied the effectiveness of our approach with and without the pruning techniques. Fig. 4 (a) shows that the NMI decreases inversely with GDR, where SIMp with "-n"/"-t" denotes the mining algorithm without the noise pruning/the time-overlapping pruning. The NMI of SIMp is similar to that of SIMp-t as GDR is smaller than 0.7 and outperforms the others for GDR$= 0\tilde{}1$. With analyzing the experiment result, we found that the time-overlapping pruning can help in partitioning objects with similar moving patterns but not closely located, and the noise pruning can filter the objects with certain pattern such as random walk to improve the grouping quality.



(a) Grouping Effectiveness (L_{max}=6, d=12). (b) Impact of PST Maximal Memory Length (L_{max}).

(c) Impact of GDAR. (d) Impact of Moving Distance (d).

Fig. 4. The simulation results.

In the second experiment, we studied the impact of the PST parameter L_{max}. Note that a PST with larger L_{max} can contain more moving patterns. Fig. 4 (b) shows that larger L_{max} results in better grouping quality. Therefore, the similarity score sim_p is useful in identifying a group of moving objects. GDAR controls the range in which objects are captured and location data are aggregated. Fig. 4 (c) shows that adaptively selecting a GDAR can keep good grouping quality while objects are more widely distributed.

While the moving distance is larger, objects can move across more sensor clusters, i.e. more CHs report their grouping results to the sink for further ensembling. In the last experiment, we investigated the influence of the object moving range (d) on the grouping quality. Fig. 4 (d) shows that the NMI of the ensembling result is improved as d increases. With analyzing the experiment result, we also found the variance of the NMI decreases inversely with d. Therefore, the ensembling can improve grouping quality and robustness.

V. Conclusion

In this paper, we investigated and utilized the characteristic of the group movement of objects to explore the group relationship in the OTSN. In contrast to the centralized mining, we proposed a novel distributed mining algorithm that consists of the GMPMine algorithm and the Cluster Ensembling algorithm to leverage the object moving patterns in grouping objects. We used the VMM model together with Probabilistic Suffix Tree (PST) in learning the moving patterns, as well as Highly Connected Component (HCS) for partitioning the objects in the GMPMine algorithm. In the Cluster Ensembling algorithm, we utilized Jaccard as the similarity measurement and NMI to evaluate the partition result in order to achieve better grouping quality. Our experiment results showed that our algorithm can effectively distinguish groups of objects and achieve good grouping quality and robustness.

References

[1] S. Ma, S. Tang, D. Yang, T. Wang, and J. Han, "Combining clustering with moving sequential pattern mining: A novel and efficient technique," *8th PAKDD*, pp. 419–423, 2004.

[2] S.-Y. Hwang, Y.-H. Liu, J.-K. Chiu, and E.-P. Lim, "Mining mobile group patterns: A trajectory-based approach," *9th PAKDD*, 2005.

[3] V. S. Tseng and K. W. Lin, "Mining temporal moving patterns in object tracking sensor networks," *Int. Workshop on Ubiquitous Data Manag.*, 2005.

[4] D. Ron, Y. Singer, and N. Tishby, "Learning probabilistic automata with variable memory length," *7th annual Conf. on Computational learning theory*, Jul. 1994.

[5] G. Bejerano and G. Yona, "Variations on probabilistic suffix trees: statistical modeling and the prediction of protein families," *Bioinformatics*, vol. 17, no. 1, pp. 23–43, 2001.

[6] S. Dubnov, G. Assayag, O. Lartillot, and G. Bejerano, "Using machine-learning methods for musical style modeling," *IEEE Computer Magazine*, vol. 36, no. 10, pp. 73–80, Oct. 2003.

[7] P. S. S. Chawla and B. Arunsalam, "Mining for outliers in sequential databases," *6th SIAM Int. Conf. onData Mining (SDM06)*, Apr. 2006.

[8] G. Mazeroff, V. D. Cerqueira, J. Gregor, and M. Thomason, "Probabilistic trees and automata for application behavior modeling," *41st ACM Southeast Regional Conf. Proceedings*, 2003.

[9] A. Apostolico and G. Bejerano, "Optimal amnesic probabilistic automata or how to learn and classify proteins in linear time and space," *Proc. of ACM RECOMB*, pp. 25–32, 2000.

[10] A. Strehl and J. Ghosh, "Cluster ensembles - a knowledge reuse framework for combining partitionings," *Proc. Conf. on Artificial Intelligence (AAAI02)*, p. 93ąV98, Jul. 2002.

[11] D. Greene, A. Tsymbal, N. Bolshakova, and P. Cunningham, "Ensemble clustering in medical diagnostics," *Proc. of 17th IEEE Symp. on Computer-Based Medical Systems (CBMS04)*, pp. 576–581, 2004.

[12] D. S. Frossyniotis, C. Pateritsas, and A. Stafylopatis, "A multi-clustering fusion scheme for data partitioning," *Int. J. Neural Syst.*, vol. 15, no. 6, pp. 391–401, Oct. 2005.

[13] D. Bollegala, Y. Matsuo, and M. Ishizuka, "Measuring semantic similarity between words using web search engines," *16th int. conf. on World Wide Web (WWW 07)*, pp. 757–766, 2007.

[14] C. P. Cheng, G. T. Lau, J. Pan, K. H. Law, and A. Jones, "Domain-specific ontology mapping by corpus-based semantic similarity," *Proc. Scientific Foundations Workshop of End-to-End Service Utility (E2ESU)*, Mar. 2007.

[15] K. W. Church and P. Hanks, "Word association norms, mutual information, and lexicography," *Computational Linguistics*, vol. 16, no. 1, pp. 22–29, 1990.

[16] J. P. W. Pluim, J. B. A. Maintz, and M. A. Viergever, "Mutual information based registration of medical images: a survey," *IEEE Trans. Med. Imaging*, vol. 22, no. 8, pp. 986–1004, Aug. 2003.

[17] R. Begleiter, R. El-Yaniv, and G. Yona, "On prediction using variable order markov models," *Journal of Artificial Intelligence Research (JAIR)*, vol. 22, pp. 385–421, 2004.

[18] C. Largeron-Leténo, "Prediction suffix trees for supervised classification of sequences," *Pattern Recogn. Lett.*, vol. 24, no. 16, pp. 3153–3164, 2003.

[19] H.-P. Tsai, D.-N. Yang, W.-C. Peng, and M.-S. Chen, "Exploring group moving pattern for an energy-constrained object tracking sensor network," *11th Pacific-Asia Conference on Knowledge Discovery and Data Mining (PAKDD 2007)*, May 2007.

[20] G. SAPORTA and G. YOUNESS, "Comparing two partitions: some proposals and experiments," *Proc. Comput. Statist.*, 2002.

[21] A. L. Fred and A. Jain, "Combining multiple clusterings using evidence accumulation," *IEEE Trans. Pattern Analysis and Machine Intelligence*, vol. 27, no. 6, pp. 835– 850, Jun. 2005.

[22] D. Bolier, "Sim : a c++ library for discrete event simulation," Oct. 1995. [Online]. Available: http://www.cs.vu.nl/ eliens/sim/

[23] B. Gloss, M. Scharf, and D. Neubauer, "Location-dependent parameterization of a random direction mobility model," *IEEE 63rd Conf. on Veh. Technol.*, vol. 3, pp. 1068–1072, 2006.