Motif Discovery in Spatial Trajectories using Grammar Inference

Tim Oates[†] Arnold P. Boedihardjo[Φ] Jessica Lin[Ж] Crystal Chen[Φ] Susan Frankenstein[Φ] Sunil Gandhi[†]

[†]University of Maryland Baltimore County
Dept. of Computer Science
oates@cs.umbc.edu
sunilga1@umbc.edu

[Φ]U.S. Army Corps of Engineers
Engineer Research and Development Center
{arnold.p.boedihardjo, crystal.chen,
susan.frankenstein}@usace.army.mil

[Ж]George Mason University
Dept. of Computer Science
jessica@gmu.edu

ABSTRACT

Spatial trajectory analysis is crucial to uncovering insights into the motives and nature of human behavior. In this work, we study the problem of discovering motifs in trajectories based on symbolically transformed representations and context free grammars. We propose a fast and robust grammar induction algorithm called mSEQUITUR to infer a grammar rule set from a trajectory for motif generation. Second, we designed the Symbolic Trajectory Analysis and VIsualization System (STAVIS), the first of its kind trajectory analytical system that applies grammar inference to derive trajectory signatures and enable mining tasks on the signatures. Third, an empirical evaluation is performed to demonstrate the efficiency and effectiveness of mSEQUITUR for generating trajectory signatures and discovering motifs.

Categories and Subject Descriptors

H.2 [**Database Management**]: Database Applications – *data mining, spatial databases and GIS.*

Keywords

spatial trajectory, motif discovery, grammar induction, activity recognition.

1. INTRODUCTION

Spatial trajectory data is increasingly becoming a critical part of human behavior analysis. From traces of GPS (Global Positioning System) signals recorded on smartphones to object movements tracked in surveillance video feeds, the modes by which spatial movements are captured have increased in both breadth and fidelity. As a result, there is a tremendous amount of data that can be used to help bring insights into the motives and behaviors of moving agents. Some examples of analytics performed on spatial trajectories are activity recognition [25], path clustering [9], and motif discovery [10].

Two critical factors that impact the performance and accuracy of trajectory analytics are the data's massive size and the presence of noise. For example, in the GeoLife data [25-27] where approximately 150 users' GPS coordinates are recorded (some for several years), a trajectory set can consist of up to 100,000 time-indexed spatial coordinates for an individual. With more than 150 users, the data set contains almost 25 million spatial points. The recorded spatial points also contain errors due to factors such as signal attenuation and computational constraints placed on the GPS devices. Hence, it is imperative that the analytical tasks employ algorithms that can efficiently and robustly process large and noisy data sets.

In this work, we study the problem of generating a compact and scalable model for large and noisy trajectory data and apply the model to discover repeated patterns (motifs). Under the assumption that observations of a spatial trajectory arise from a generative process that probabilistically outputs trajectory segments across time, a feature model that can embed this essential characteristic and meets the constraints for scalability and compactness is a grammar rule set. Hence, we address the voluminous and noisy data issues by transforming the 2D or 3D spatiotemporal data into a symbolic representation and modeling it via grammar rules. This paper proposes a fast and robust induction algorithm called mSEQUITUR to generate the grammar rules and discover motifs in spatial trajectories. In addition, we developed the Symbolic Trajectory Analysis and VIsualization System (STAVIS), the first of its kind trajectory analytical system that applies our signature representation to a variety of trajectory analytical operations. This system supports an end-to-end analytical framework for querying, processing, and visualizing symbolic spatial trajectories.

The major contributions of this paper are summarized as follows:

1. Proposed a novel linear space/time grammar induction (GI) algorithm, mSEQUITUR, for spatial trajectories that is robust to noise and effectively models the geometric relationships of the trajectories.
2. Developed the first analytical system framework, STAVIS, for symbolic spatial trajectories that supports spatiotemporal queries, pattern mining, and visualization.

2. BACKGROUND AND RELATED WORK

In this section, we briefly discuss background and related work on GI. We begin by defining our data type of interest, spatial trajectories, and the problem statement:

Definition 1. *Spatial trajectory*: A trajectory t is a time indexed and ordered set of (noisy) location points sampled from a curve generated by an object moving in 2-dimensional geographic space.

Problem Statement. Given a trajectory t, generate a robust and compact grammar rule set from t with time and space complexity linear in the size of t.

Our goal is to build models of trajectory data that facilitate both deep human understanding of the underlying generative processes and additional machine processing for motif detection. The choice of model class is thus extremely important because it must be expressive, a good match to the structure in the data, and amenable to human inspection. In addition, our application domain demands efficient learning. Grammars meet all of these desiderata.

GI is the problem of identifying a grammar from a target class using a set of strings known to belong to the language of the grammar (positive examples) and, optionally, a set of strings known to not belong to the language (negative examples). At a coarse level, GI methods tend to be based on merging or splitting [5]. In the latter case, the initial grammar accepts all possible strings and is refined (made more specific) by creating new non-terminals and rules [24]. Merging approaches start with a grammar that accepts all of the positive examples and no other strings [22]. Non-terminals and rules are then *merged* to generalize the language to accept unseen strings (but no negative examples if they are available).

Our work is based on the SEQUITUR algorithm [19], which learns context-free grammars in time O(n) where n is the size of the input. In [11], we proposed a grammar-based motif discovery algorithm for time series, and SEQUITUR was the choice of grammar induction algorithm. The experimental results show that the adaptation of SEQUITUR to time series data successfully discovers repeated patterns of previously unknown lengths. While the results are promising, there are some limitations with the algorithm. More specifically, SEQUITUR uses a splitting operator to replace

CIKM' 13, Oct. 27-Nov. 1, 2013, San Francisco, CA USA.

http://dx.doi.org/10.1145/2505515.2507820

repeated subsequences in the input string with a non-terminal and, when applied recursively, leads to rule hierarchies. Because there is no merging operator, the resulting grammar does not generalize – it only accepts the given input string. As a result, it does not work well for noisy data. We address this major weakness in this paper. Our approach to merging is related to that employed by Bayesian model merging [20], though the goal there is to balance a description length prior on grammars with the probability of the data given the grammar, and is most similar to previous work on learning reversible regular [1] and context-free [20] grammars that make extensive use of contexts.

From the perspective of spatial trajectory mining, the majority of existing work concentrates on indexing and similarity searchs in large spatial databases [3], most of which are based on shape-based similarity measures. Other pattern learning tasks include periodicity analysis [12], relative motion identification [6], trajectory classification [8], spatial co-occurrence discovery [2], clustering [9], and outlier detection [7, 10]. These techniques were shown to be useful for discovering animal and bird migration patterns, patterns in hurricane tracks, and abnormalities in vehicle movement data. Analytical and visualization systems have also been developed for spatial trajectories [18]. However, most of these systems focus on cluster analysis and are optimized for player tracking in recreational sports.

To the best of our knowledge, our work is the first to apply GI to the problem of efficiently generating signatures of spatial trajectories. The signatures are applied within a symbolic system framework, STAVIS, to improve the robustness of trajectory pattern analysis.

3. Model Generation

We propose a three-step transformation process that allows efficient computation and adaptation of grammar induction algorithms on spatial trajectories to generate the grammar rule set as follows.

Step 1 (Data Linearization): We employ a space-filling curve (SFC) to map the spatial trajectory data into one dimensional data indexed by time. Space-filling curves provide an efficient and effective way to linearize data such that spatial locality is preserved. They have been used extensively for multidimensional data organization and for spatial data partitioning [16]. We have selected the Hilbert space-filling curve due to its superior ability to preserve distances [17].

Step 2 (Data discretization): Once we linearize the trajectory, we discretize the series into a symbolic string sequence using Symbolic Aggregate approXimation (SAX) [14]. SAX offers many unique advantages over other aggregation methods. Specifically, SAX allows dimensionality reduction, which not only improves computational complexity, but also removes noise and, consequently, produces more meaningful results.

Step 3 (Grammar Induction): We infer a grammar from the SAX string produced from Step 2 using mSEQUITUR. Each string delimited by a space represents one or more consecutive subsequences and is treated as a terminal symbol, an atomic unit for patterns. mSEQUITUR generates a hierarchy of rules, each of which represents a repeated pattern, and from these rules, additional machine processing (e.g., motif discovery) can be performed.

3.1 mSEQUITUR

The starting point for our work is the SEQUITUR algorithm for inferring context-free grammars from sequences. The algorithm processes the input sequence in a single left-to-right pass and produces as output a context-free grammar that generates one string, the input sequence. SEQUITUR does this by enforcing two constraints: bigram uniqueness and rule utility. The first constraint requires that every bigram occurs only once in the grammar, and is violated when the second occurrence of a bigram is seen in the left-to-right pass over the input sequence. To enforce this constraint the two occurrences of the bigram are replaced by a non-terminal that expands to the bigram. The other constraint enforced by SEQUITUR is rule utility, which requires that every non-terminal be used more than once. Over time the interaction of these two rules exposes recurring hierarchical structure in the input sequence.

As stated earlier, though SEQUITUR is exceptionally efficient for a GI method, it does not generalize due to the lack of a merging operator. That means the resulting grammars cannot generate or recognize any sequences not provided as input. Practically, grammars trained on suspicious trajectories would only recognize those trajectories, and not other suspicious trajectories that have similar, but not identical, structure. In this section we describe mSEQUITUR, a major extension of SEQUITUR that allows merging yet is still efficient.

The merging operator is described as follows. Given two non-terminals, X and Y, create a new non-terminal Z and replace all occurrences of X and Y in the grammar with Z. The resulting grammar can now generate new strings, ones in which a substring generated by an X is replaced by a substring generated by a Y, and vice versa. The difficult question is deciding when such a merge is warranted. The usual answer is to look at *context*.

Given a non-terminal A, let $L_d(A)$ be the set of strings generated from A by a tree with depth bound *d*. Note that these strings may contain both terminal and non-terminal symbols. Given $s \in L_d(A)$, let C(s) be the contexts of s, which is a set of strings whose size is the same as the number of non-terminals in s. The elements of s are created by making a copy of s and replacing one non-terminal with the wildcard symbol *. The resulting wildcarded string is a context, with the wildcard marking a non-terminal position and the remainder of the string being the context. If two different non-terminals can fill the wildcard position in the same context (i.e., if such a string actually occurs), then those non-terminals are candidates for merging.

Consider the following simple example. Given the input sequence abcababc, SEQUITUR produces the following grammar:

$S \rightarrow R1\ R2\ R1;\ R1 \rightarrow R2\ c;\ R2 \rightarrow a\ b$

The start symbol, S, expands to the string of non-terminals R1 R2 R1. Non-terminal R1 expands to a right-hand side with one non-terminal (R2) and one terminal (c). Non-terminal R2 expands to the terminal string ab. If we expand the start symbol to depth 1 we get the string R1 R2 R1. Expanding to depth 2 yields the string R2 c a b R2 c, and expanding to depth 3 gives the string abcababc. As the expansion depth increases the strings become longer and tend to have more terminals and fewer non-terminals.

Given string s = R1 R2 R1, the contexts of s, denoted C(s), are {* R2 R1, R1 * R1, R1 R2 *}. Given a fixed depth d, we generate $L_d(A)$ for all non-terminals A, and then the contexts of each string generated. Note that s = R1 R2 R1 is in $L_1(S)$. If any two different fully instantiated (non-wildcarded) string match a context (where any non-terminal can match a wildcard), then the non-terminals that match the wildcard position are merge candidates.

When dealing with spatiotemporal trajectories, a context is a partial trajectory. That is, the terminals and non-terminals in the sequence ultimately expands to a sequence of terminals, and by virtue of the way the terminals were generated and the grammar was learned, the terminal sequence represents the path through space and time at the level of the raw trajectory data. Note that every non-terminal in the original SEQUITUR algorithm expands to exactly one string of terminals. When two strings differ in precisely one position that contains a non-terminal, it means that they correspond to two

terminal strings that share some structure but differ in the trajectory generated at the unmatched position.

Time and space complexity: One of the appealing features of SEQUITUR is its efficiency. Therefore, our goal is to support merging, and thus learning of grammars that generalize, efficiently. In particular, our proposed mSEQUITUR generates grammar in time linear in the input sequence length n. We first note that SEQUITUR is a bottom-up algorithm that runs in time linear in the size of the input sequence. To support merging in mSEQUITUR, we need a list of the strings that can be generated by each non-terminal to a fixed depth and the contexts (wildcarded strings) associated with each. Because the algorithm is bottom-up, it is trivial to augment the representation to include the depth of a non-terminal, which is the depth of the tree it generates. When forming non-terminals that expand directly to strings of terminals, their sole depth 1 expansion is recorded. When forming a depth 2 non-terminal, its full expansion can be recorded based on the expansions of its right-hand side. By recording with each non-terminal its deepest expansion to depth d−1, it is possible to maintain information about strings generated to depth d while learning the grammar with only constant overhead.

Given a non-terminal's depth d expansion, the contexts for that string can be generated in a linear pass over the expansion and stored in a hash table keyed on context where the value stored is a list of non-terminals that fill wildcard positions. This hash table makes it possible to identify all merge candidates in time O(nd) for input sequence length n and depth d expansions of non-terminals. Note also that the space required to store the contexts is linear in the size of the input. Initially, each non-terminal generates exactly one string to depth d, and the contexts are represented implicitly by the positions of the non-terminals in the contexts. Merging does not generate any new contexts, though it does allow them to be combined in new ways when making future merging decisions. Therefore, while mSEQUITUR runs, the space requirements do not grow, but remain linear in the size of the input.

3.2 STAVIS

STAVIS supports an end-to-end analysis workflow through a web-based platform. It includes a spatial data repository that stores recorded trajectory traces composed of the following features: track ID, spatial point coordinates, and times. Although a trajectory is a continuous function mapping from the time domain to the spatial domain, in practice they are recorded as discrete spatiotemporal (ST) point samples. As a result, the database represents a trajectory as an ST point set.

STAVIS integrates three primary functions: *ST filtering*, *trajectory transformation*, and *signature analysis and visualization*. The trajectories of interest are obtained through *ST filtering* by performing an ST query on the dataset. *Trajectory transformation* applies the Hilbert space-filling curve and SAX to map the ST points to a time-indexed string sequence. *Signature analysis and visualization* apply grammar induction (mSEQUITUR and SEQUITUR) on the SAX representations and perform motif discovery and visualization on the grammar rules.

STAVIS implements SEQUITUR and mSEQUITUR for signature generation and utilizes the resulting grammar to enable the following motif discovery methods:

Fixed-length motif discovery: applied directly on SAX words and employs fixed-length motif finding algorithm [13, 21]; **Variable-length motif discovery:** applied on signatures generated by SEQUITUR and extracts motifs from the signatures; **Noise tolerant variable-length motif discovery:** applied on signatures generated by mSEQUITUR and extracts motifs from the signatures.

4. EXPERIMENT

To evaluate the efficiency of our proposed mSEQUITUR method and symbolic based approach to trajectory data, we compare the baseline motif [13, 21], SEQUITUR, and mSEQUITUR approaches to determine their capacity to identify patterns or anomalies in various scenarios using STAVIS. We compiled two different test datasets from Microsoft's GeoLife and a synthetically generated trajectory using [4, 23]. The GeoLife trajectories were concatenated to mimic repetitive trips with slight divergences. The synthetic dataset, called DC_Synth, is a trajectory with approximately 10,000 sampled points and generated by concatenating one trajectory to itself four times to achieve known repetition.

For each of these datasets, we look at the original trajectory on the map to visually identify where common patterns are expected to occur. We also pay attention to locations where discontinuities occur on the Hilbert SFC transformed representation to analyze the impact this may have on the discovered patterns. Next, the baseline motif, SEQUITUR, and mSEQUITUR are performed on the SAX transformed Hilbert curve and the resulting patterns compared. If a specific pattern appears dissimilar to a related occurrence, we confirm the results by referring to the time series graph to help understand why the two seemingly dissimilar subsequences were categorized into the same pattern. Furthermore, we compare each algorithm (baseline motif, SEQUITUR, and mSEQUITUR) to see if new patterns are discovered or if previously matching patterns are no longer being captured or matched together.

We experimented with different parameters to evaluate the system's sensitivity to parameter choices. Initially, we started with the parameters used in [14, 15] that were found to be most effective in discovering and matching the patterns. Then we made slight changes to those values to avoid largely overlapping subsequences that hindered visualization, but still allowed for the discovery of the expected patterns. It was found that SAX window size = 20, alphabet size = 4, Hilbert grid order = 2^4, and mSEQUITUR depth = {1,2} gave us the best results. Additionally, the use of a sliding window allows us to capture every possible subsequence in the time series, while the use of numerosity reduction allows us to reduce overlap and detect variable length patterns.

In Fig. 1, red points on the map represent the original trajectory points and blue points show the entire subsequence of a pattern. Annotations on the time series graphs show starting points of pattern subsequences. A solid lined box displays the location of a discovered rule occurrence, and a dashed box of the same color indicates where the matching rule occurrence was expected but not found. Fig. 1 (a-c) illustrate three different rules that SEQUITUR has generated, Fig. 1(d) shows the results of baseline motif, and Fig. 1(e) depicts the patterns discovered by mSEQUITUR. Although Fig. 1(a) and Fig. 1(b) detect a frequent pattern on one half of the trajectory, they both fail to detect its corresponding pattern on the other repeated, half of the trip. The rule depicted in Fig. 1(c), however, is able to find the frequent patterns that were missed in Fig. 1(a) and Fig. 1(b). All of these rules, however, were successfully captured by mSEQUITUR's context and merge candidates shown in Fig. 1(e). Since mSEQUITUR matches the similar contexts of these patterns and captures the potential merge candidates, the combination of all three related rules results in more discoveries of the expected pattern occurrences. For the baseline motif results (Fig. 1(d)), the algorithm discovered many of the patterns that the three SEQUITUR rules missed, but still misses some of the patterns that mSEQUITUR was able to detect. While mSEQUITUR is not always able to detect every matching pattern, it is able to perform as efficiently as SEQUITUR and the baseline motif but with higher accuracy.

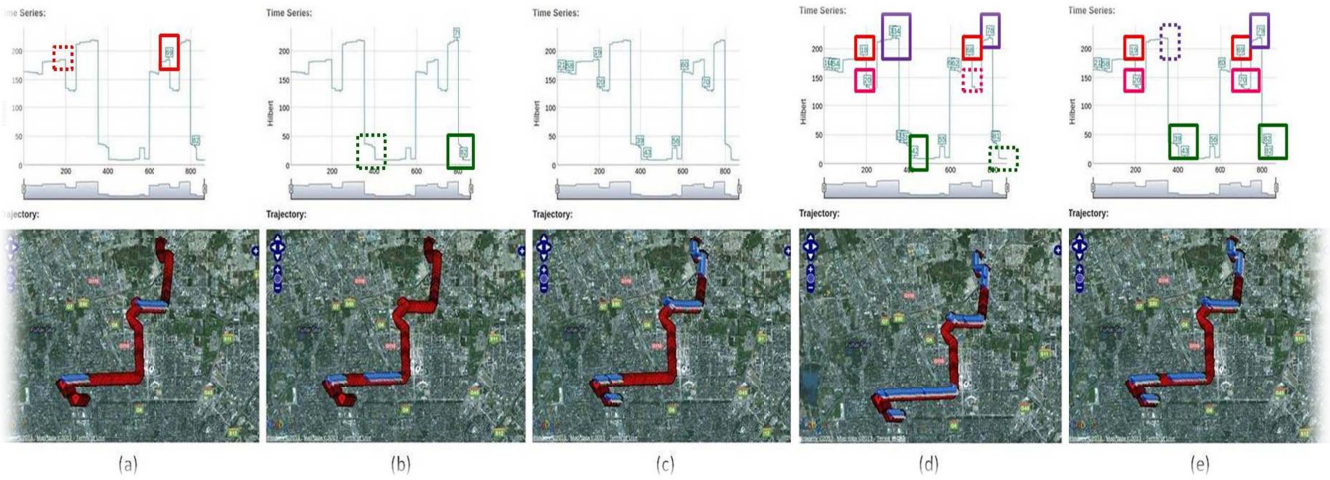

Figure 1: Comparison between rules discovered by SEQUITUR (images a, b, and c), one of the patterns discovered through baseline motif (image d), and those found with mSEQUITUR (image e).

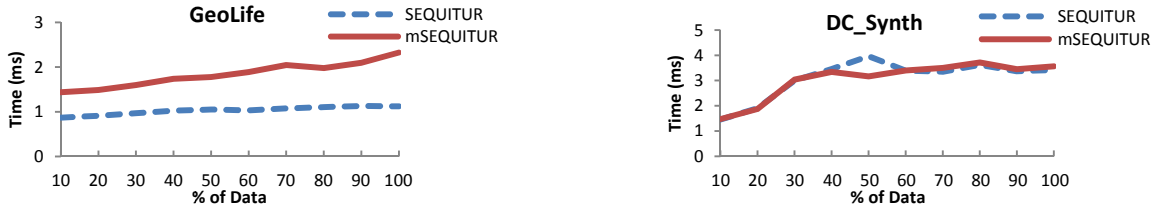

Figure 2: SEQUITUR and mSEQUITUR timing results for GeoLife and DC_Synth trajectories.

Fig 2. shows the runtimes of mSEQUITUR and SEQUITUR on varying sizes of the selected GeoLife and DC_Synth datasets. The figure gives the average of 22 runs with the 2 highest and 2 lowest runs discarded. The experimental platform was an Intel Core i5 @ 2.50GHz with 6GB RAM running Ubuntu 10.10. In both datasets, mSEQUITUR's performance was comparable to SEQUITUR and matched the theoretical results of the linear order cost provided in Section 3.1. For DC_Synth at the 30% mark, there is a higher than average increase in the running times for both algorithms which is attributable to the additional patterns that were discovered. However, the upper bound of their running times is still linear to the input size. The experiment shows that the mSEQUITUR algorithm can efficiently discover patterns (in linear time) for spatial trajectory data.

5. CONCLUSION AND FUTURE WORK

In this paper, we proposed the mSEQUITUR algorithm and STAVIS framework to effectively generate signatures and support motif discovery for spatial trajectories. Due to mSEQUITUR's ability to generalize grammar rules, it is able to discover the relevant motifs in noisy trajectories and provide higher recall rates than the competing methods. As future work, the mSEQUITUR algorithm will be extended to tri-grams and consider additional geometric constraints to meet different application requirements.

6. REFERENCES

[1] D. Angluin, "Inferences of Reversible Languages," *Journal of the ACM,* vol. 29, pp. 741-765, 1982.

[2] M. Celik, *et al.*, "Mixed-Drove Spatio-Temporal Co-occurrence Pattern Mining," *IEEE Transactions on Knowledge and Data Engineering,* vol. 20, pp. 1322 - 1335 2008.

[3] C. Costa, *et al.*, "Finding Similar Trajectories in Smartphone Networks without Disclosing the Traces," in *International Conference on Data Engineering*, 2011.

[4] C. Duntgen, *et al.*, "BerlinMOD: A Benchmark for Moving Object Databases," *Very Large Databases Journal,* vol. 18, pp. 1335-1368, 2009.

[5] C. D. L. Higuera, *Grammatical Inference: Learning Automata and Grammar*: Cambridge University Press, 2010.

[6] P. Laubea, *et al.*, "Discovering Relative Motion Patterns in Groups of Moving Point Objects," *International Journal of Geographical Information Science,* vol. 19, pp. 639-668, 2005.

[7] J.-G. Lee, *et al.*, "Trajectory Outlier Detection: A Partition-and-Detect Framework," in *International Conference on Data Engineering*, 2008.

[8] J.-G. Lee, *et al.*, "TraClass: Trajectory Classification Using Hierarchical Region-based and Trajectory-based Clustering," in *Proceedings of Very Large Databases*, 2008, pp. 1081-1094.

[9] J.-G. Lee, *et al.*, "Trajectory Clustering: A Partition-and-Group Framework," in *ACM SIGMOD Conference on Management of Data*, 2007.

[10] X. Li, *et al.*, "ROAM: Rule-and Motif-Based Anomaly Detection in Massive Moving Object Data Sets," in *SIAM International Conference on Data Mining*, 2007.

[11] Y. Li, *et al.*, "Visualizing Variable-Length Time Series Motifs," in *SIAM International Conference on Data Mining*, 2012.

[12] Z. Li, *et al.*, "Mining Hidden Periodic Behaviors for Moving Objects," in *ACM SIGKDD Conference on Knowledge, Discovery, and Data Mining*, 2010.

[13] J. Lin, *et al.*, "Visualizing and Discovering Non-trivial Patterns in Large Time Series Databases," vol. 4, pp. 61-82, 2005.

[14] J. Lin, *et al.*, "Experiencing SAX: A Novel Symbolic Representation of Time Series," *Data Mining and Knowledge Discovery,* vol. 15, pp. 107-144, 2007.

[15] J. Lin, *et al.*, "Finding Structural Similarity in Time Series Data Using Bag-of-Patterns Representation," in *International Conference on Scientific and Statistical Database Management*, 2009, pp. 461-477.

[16] L. Meng, *et al.*, "An Improved Hilbert Curve for Parallel Spatial Data Partitioning," *Geo-Spatial Information Science Journal,* vol. 10, pp. 282-286, 2007.

[17] B. Moon, *et al.*, "Analysis of the clustering properties of the Hilbert space-filling curve," *IEEE Transactions in Knowledge and Data Engineering,* pp. 124-141, 2010.

[18] C. Mutschler, "Online Data - Mining of Interactive Trajectories in Realtime Location Systems," Friedrich-Alexander-University of Erlangen-Nuremberg, 2010.

[19] C. G. Nevill-Manning, *et al.*, "Identifying Hierarchical Structure in Sequences: A Linear-time Algorithm," *Journal of Artificial Intelligence Research,* vol. 7, pp. 67-82, 1997.

[20] T. Oates, *et al.*, "Learning k-Reversible Context-Free Grammars from Positive Structural Examples," in *Proceedings of the International Conference on Machine Learning*, 2005, pp. 459-465.

[21] P. Senin. (2013). *JMOTIF Time Series Mining: A Time Series Data-Mining Toolkit based on SAX and TFIDF Statistics*. Available: http://code.google.com/p/jmotif/

[22] A. Stolke, *et al.*, "Inducing Probabilistic Grammars by Bayesian Model Mergin," in *Grammatical Inference and Applications*, 1994, pp. 106-118.

[23] S. Yackel, *et al.* (2013). *Minnesota TG: Web-based U.S. Road Traffic*. Available: http://mntg.cs.umn.edu

[24] M. V. Zaanen, *et al.*, "Model Mergin versus Model Splitting Context-Free Grammar Induction," *Journal of Machine Learning Research - Proceedings Track 21,* pp. 224-236, 2012.

[25] Y. Zheng, *et al.*, "Understanding Mobility Based on GPS Data," presented at the ACM Conference on Ubiquitous Computing, 2008.

[26] Y. Zheng, *et al.*, "GeoLife: A Collaborative Social Networking Service among User, Location, and Trajectory.," *IEEE Data Engineering Bulletin,* vol. 33, pp. 32-40, 2010.

[27] Y. Zheng, *et al.*, "Mining Interesting Locations and Travel Sequences from GPS Trajectories," in *International Conference on World Wide Web*, 2009, pp. 791-800.