INTERNATIONAL JOURNAL OF RESEARCH IN COMPUTER APPLICATIONS AND ROBOTICS
ISSN 2320-7345

DETECTING THE PERIODIC OUTLIER PATTERN USING TIME SERIES SEQUENCES

Ms.M.LATHA[1], Ms.P.SUGUNADEVI[2] *M.E, AP/CSE.*

[1]*M.E COMPUTER SCIENCE AND ENGINEERING DEPARTMENT ANNA UNIVERSITY CHENNAI,*

PGP COLLEGE OF ENGINEERING AND TECHNOLOGY, NAMAKKAL, TAMILNADU, INDIA
Email address:lathamecse91@gmail.com

[2]*ASSISTANT PROFESSOR/COMPUTER SCIENCE AND ENGINEERING DEPARTMENT, ANNA UNIVERSITY CHENNAI, PGP COLLEGE OF ENGINEERING AND TECHNILOGY, NAMAKKAL, TAMILNADU, INDIA*
Email address:sugunadevime@gmail.com

Abstract

Detecting the periodicity of outlier patterns might be more important in many sequences than the periodicity of regular, more frequent patterns. In this paper, Ipresent the development of a enhanced suffix array-based time efficient algorithm for unusual or outlier periodic patterns. The development of a mathematical model to measure how unusual or surprising a pattern is compared with other patterns in the same data sequence;The ability to detect periodic patterns that appear in a subsection of the series and extensive comparative experimental evaluation of various aspects of the algorithm has been conducted, and it has been favorably compared with Info Miner**.**

Index Terms—Outlier periodic patterns, performance, periodicity detection, suffix tree, surprising patterns, surprising periodicity, time series, unusual periods.

I. Introduction

Time series data accounts for an increasingly large fraction of the world's supply of data. A random sample of 4,000 graphics from 15 of the world's newspapers published from 1974 to 1989 found that more than 75% of all graphics were time series (Tufte, 1983). Given the ubiquity of time series data, and the exponentially growing sizes of databases, there has been recently been an explosion of interest in time series data mining. In the medical domain alone, large volumes of data as diverse as gene expression data electrocardiograms, electroencephalograms, gait analysis and growth development charts are routinely created.Similar remarks apply to industry, entertainment, finance, meteorology and virtually every other field of human endeavor. Although statisticians have worked with time series for more than a century, many of their techniques hold little utility for researchers working with massive time series databases.Below are the major task considered by the time series data mining community.

Indexing (Query by Content): Given a query time series Q, and some similarity/dissimilarity measure D (Q;C), find the most similar time series in database DB

Clustering: Find natural groupings of the time series in database DB under some similarity/dissimilarity measure D (Q;C)

Classification: Given an unlabeled time series Q, assign it to one of two or more predefined classes

*Prediction (Forecasting):*Given a time series Q containing n data points, predict the value at time n + 1.

Summarization: Given a time series Q containing n data points where n is an extremely large number, create a (possibly graphic) approximation of Q which retains its essential features but fits on a single page, computer screen, etc.

Segmentation:

(a) Given a time series Q containing n data points, construct a model ^1Q, from K piecewise segments (K << n), such that Q closely approximates Q (Keogh and Pazzani, 1998).

(b) Given a time series Q, partition it into K internally homogenous sections (also known as change detection Note that indexing and clustering make explicit use of a distance measure, and many approaches to classification, prediction, association detection, summarization, and anomaly detection make implicit use of a distance measure.

II. Related Work

Outlier

Outlier detection focusing on finding some interesting patterns that out of norm. According to [1], anomalous patterns also referred as outlier, anomalies, discordant observations, exceptions, faults, defects aberrations, noise, errors, damage, surprise, novelty, peculiarities or contaminants in various applications domain. In such for public health, outlier detection being widely used to detect anomalous patterns in patient records which could be symptoms or disease.

Outlier Detection

Of all the data mining techniques that are in vogue, Outlier Detection comes closest to the metaphor of mining for nuggets of information in large databases. My working definition of outlier detection is slightly more broad than what is conventionally accepted. For example, I consider mining for rare association rules (low support and high confidence), classification for imbalanced data sets and of course the use of non-parametric techniques (e.g., distance and density based) to find isolated entities as examples of Outlier Detection.

Periodicity is a commonly seen phenomenon in moving objects as well as social networks. People usually go to workplace every day through more or less same route, birds and animals migration from one place to another also show yearly periodicity. Interactions between people and topic discussion in social media also show periodicity, for example delivering news about different budgets in news sites and posting of information corresponding to different events like annual conferences in blogs and websites are done periodically. The periodic pattern mining algorithms designed for event symbol sequences cannot be used to find spatiotemporal periodic patterns. This is because the repetitions of spatial locations may not be identical. Even if objects follow same route regularly, they may not appear at exactly the same location regularly. For example a person may go from home to office in the same route every day between 9.00 A.M. to 10.00 A.M. But it is less likely that he will be at the same location on his route everyday at 9.30 A.M. [2].Cao el al. have proposed an algorithm to find maximal periodic patterns from spatiotemporal data [3].

This algorithm mines periodic 1-patterns using spatial clustering. Then bottom-up and top-down mining techniques are used for generating longer patterns. They have also proposed algorithms which can mine periodic patterns that appear in only a time interval instead of whole time span. Another algorithm which can mine shifted or distorted instances of patterns is also proposed. Algorithm for mining and indexing of spatiotemporal periodic patterns from historical spatiotemporal data was developed by N. Mamoulis el al [4]. Z. Li el al. define a periodic behavior as repeating activities at certain locations with regular time intervals. A two stage algorithm called Periodica [5] is used to detect periods and find the periodic behaviors. In the first stage reference spots which are dense regions that are frequently visited in the movement are identified. Then the periods associated with each reference spot are identified using Fourier transform and autocorrelation [6]. Since every period is associated with a reference spot, if I find the periods associated with each reference spot we can guarantee that all the periods in the movement are detected. A period may be associated with two or more reference spots. In the second stage all the reference spots associated with a period are considered together for mining the periodic behaviors.

A periodic behavior is a statistical description of the periodic movement for a period. It is a probability distribution matrix which gives the probability that the object is at each reference spot at different sub intervals of a period. For example when we mine periodicities of a school student I may observe 24 hours period at three reference spots namely class room and play ground and house at different hours of the day. The periodic behavior for the student gives the probability with which the student is at different reference locations namely class room, playground and house in each hour of the day. In [7] the authors have extended this work by explaining how periodic behaviors can be used for missing data interpolation and future movement prediction. Z. Li et al, have developed probabilistic model to mine periodic patterns from noisy and incomplete observations [8]. Mobile, GPS and Sensor technologies help us to track human and animal movements. But the data collected by these devices have a large portion of missing and noisydata. The data is also unevenly sampled and the rate of sampling in some cases like data sent by sensors attached to animal is very low. Fourier transforms and autocorrelation generally require evenly sampled data collected at high sampling rate. So these methods fail in this case. So a new probabilistic model is proposed to mine periodic patterns in these types of datasets.

In this paper, I present a periodic outlier pattern detection algorithm that grants more significance to the unusual patterns. The algorithm uses suffix tree as the underlying data structure, and uses comparative repetitions of a pattern to measure how unusual it is comparatively. The proposed algorithm is designed on top of our previous algorithm suffix tree-based noise resilient (STNR) algorithm [9]. Contrary to STNR, the proposed algorithm focuses specifically on outlier or unusual patterns. Here are several algorithms that discover the frequent periodic patterns having (user specified) minimum number of repetitions or with minimum confidence (ratio between number of occurrences found and maximum possible occurrences), However, not much work has been done for periodicity detection of outlier patterns. It is important to note that surprising, unusual, or outlier patterns.

Periodicity Detection for Outlier Patterns

In this section, it explains the process of periodic outlier pattern mining. The process can be summarized in the following steps

Build a suffix tree for the input sequence;

Suffix tree is a commonly used data structure that has been proved very useful in string processing. It can be efficiently used to find a substring in the original string, to find the frequent substring and other string matching problems. A suffix tree for a string represents all its suffixes. It contains a distinguished path from the root for each of the suffixes of the string. The most important aspect of the suffix tree related to our work is that it very efficiently captures and highlights the repetitions of substrings within a string. Fig. 1 shows a suffix tree for the string *abcabbabb*$, where $ denotes the end marker for the string, a unique symbol that does not appear anywhere in the string (also called a sentinel).

Annotate the enhanced suffix array

Once the tree is constructed, we annotate the tree such that each internal node has two pieces of information: 1) the length of the substring it represents and 2) the number of leaves under the subtree rooted at this internal node (this represents the number of times the represented substring is repeated in the original sequence).

In order to compare the frequency of a pattern with all patterns of similar length, we build a PFT which records the number and total frequency of all l − length patterns, $1 \leq l \leq k$, where k is the maximum length of the pattern; it is computed as $k = \log\Sigma(n)$, where Σ is the number of alphabets, and n is the length of the sequence. It has already been proved that the patterns of length greater than $k = \log_{\Sigma}(n)$ are hardly repetitive. The length of the substring, which is represented by an internal node, is already calculated and recorded during the construction of the suffix tree. By performing one bottom-up traversal of the tree, we record the number of leaves under the subtree rooted at each internal node, and build the PFT. During traversal, each internal node having only leaves as children nodes records its number of leaves and passes its leaf count to the parent node. Each internal node, with some of the children being internal nodes, records the sum of the number of its leaf children and the already computed counts of its nonleaf children. Each internal node u representing substring X increments in PFT the frequency and count with its leaves count against all li , $|X| - EdgeLabel(u) \leq i \leq |X|$, where $EdgeLabel(u)$ is the label of incoming edge of node u and $|X|$ is the length of pattern X. Since the PFT is updated only at internal nodes, only patterns repeating at least twice are recorded in the PFT. Algorithm 1, Annotate Tree, presents the aforementioned process.

Candidate Outlier Patterns identification

Once the tree is annotated, each internal node has the length and the frequency of repetitions of the represented pattern. Now we are all set to detect outliers by running our previous STNR algorithm with few more constraints. We traverse the tree in bottom-up fashion until the direct children of the root are reached. At each level, every node passes its value to the

edge connecting its parent to the next level, which is called the parent edge. The parent edge collects the values from its children and creates a vector called the occurrence vector which contains from the original string the index positions at which the string from the root to that edge has repeated or occurred. Each edge also passes its occurrence vector to the parent edge; and this process recursively continues until the first level edges (the level below the root). At each intermediate edge, an edge that leads to a non leaf node, the periodicity detection algorithm checks if the string represented by the edge is periodic using its occurrence vector. Recall that in here, we are only interested in the outlier patterns; therefore, before invoking the periodicity detection procedure, we check if the pattern represented by an intermediate node (or edge) is a candidate outlier pattern and if its surprise measure is greater than the minimum surprise value. The tree traversal is implemented using the non recursive explicit stack-based algorithm, which prevents the program from throwing the stack-overflow-exception. The whole process has been reflected into Algorithm 2, which is called Mine Outlier Periodicity.

Identify the candidate outlier patterns.

For periodicity detection, we use Algorithm 3 (which includes the basics of STNR) to process the occurrence vectors. Briefly, STNR is a distance-based algorithm where a candidate period is the difference between two consecutive occurrences of a pattern. It traverses the occurrence vector once and records the test periods along with their frequency which keeps on updating as the occurrence vector is traversed. It have only presented the basic STNR algorithm to provide an idea of how STNR works; the complete algorithm (with time tolerance and maximum distance). Since outlier patterns are expected to be rare and may appear with larger period values with non-strict periodic repetitions, the time tolerance window should also be specified larger than that for frequent periodic patterns.It believe that the time tolerance window concept is handier when dealing with outlier periodic patterns. Similarly the minimum confidence value should also be set lower than that for the frequent periodic patterns.

Enhanced suffix array

The term enhanced suffix array stands for data structures consisting of a suffix array and additional tables.It will see that every algorithm that is based on a suffix tree as its data structure can systematically be replaced with an algorithm that uses an enhanced suffix array and solves the same problem in the same time complexity. The generic name *enhanced suffix array* stands for data structures consisting of the suffix array and additional tables. Our new algorithms are not only more space efficient than previous ones, but they are also faster and easier to implement. The *suffix array* suftab of the string S is an array of integers in the range 0 to n, specifying the lexicographic ordering of the $n + 1$ suffixes of the string $S\$$. That is, Ssuftab[0],Ssuftab[1], . . . ,Ssuftab[n] is the sequence of suffixes of $S\$$ in ascending lexicographic order. The suffix array requires $4n$ bytes. the space consumption of the suffix tree has been identified to be a major problem when comparing large genomes.Itwill solve this problem by using the suffix array enhanced with the lcp-table. The *lcp-table* lcptab is an array of integers in the range 0 to n. We define lcptab[0] = 0 and lcptab[i] is the length of the longest common prefix of Ssuftab[$i-1$] and Ssuftab[i], for $1 \leq i \leq n$. Since Ssuftab[n] = \$, we always have lcptab[n] = 0. The lcp-table can be computed as a by-product during the construction of the suffix array, or alternatively, in linear time from the suffix array.

Suffix Tree

Suffix tree is a commonly used data structure [10] that has been proved very useful in string processing [11]. It can be efficiently used to find a substring in the original string, to find the frequent substring and other string matching problems. A suffix tree for a string represents all its suffixes. It contains a distinguished path from the root for each of the suffixes of the string. The most important aspect of the suffix tree related to our work is that it very efficiently captures and highlights the repetitions of substrings within a string. Fig. 1 shows a suffix tree for the string abcabbabb\$, where \$ denotes the end marker for the string, a unique symbol that does not appear anywhere in the string (also called a sentinel).

The path from the root to any leaf represents a suffix for the string. Since a string of length n can have exactly n suffixes, the suffix tree for a string also contains exactly n leaves. Each edge is labeled with the string that it represents. Each leaf node holds a number that represents the starting position of the suffix yield when traversing from the root to that leaf. Each intermediate node contains a number which is the length of the substring read when traversing from the root to that intermediate node.

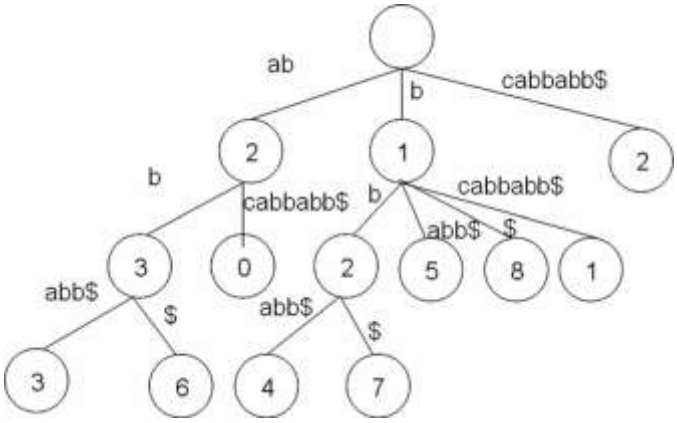

Fig.1. Suffix tree for the string abcabbabb$.

Each intermediate edge reads a string (from the root to that edge), which is repeated at least twice in the original string. These intermediate edges form the basis of our algorithm presented later in this section.

A suffix tree can be generated in linear time using Ukkonen's algorithm described in [12]; it is online algorithm, i.e., it allows extending a suffix tree as new symbols are added to the string. A suffix tree for a string of length n can have at most 2n nodes, and the average depth of the suffix tree is of the order log(n) [13], [14]. It is not necessary to always keep a suffix tree in memory; there are algorithms [15],available for the disk-based suffix trees making it a very preferred choice for processing very large sized strings such as time series [16] and DNA sequences which grow in billions [17].

III. Experimental Evaluation

I present the results of the experiments performed to test various aspects of the algorithm employed in outlier patterns detection; it is called STNR-out in the remainder of this section. The algorithm is compared with InfoMiner [18].A synthetic dataset is used to test the accuracy and time performance of the algorithm. All the experiments are compared with InfoMiner to highlight areas where STNR-out performs better than existing approaches. It also demonstrate how the time performance of the algorithm remains almost unchanged and is not affected by the number of outlier patterns detected, provided the overall series size remains the same.

It is important to note that SANR-out only highlights the surprising part of the pattern and not the events at all index positions. It argues that this is more useful than presenting the entire pattern containing both surprising and nonsurprising events (as done by InfoMiner), which makes it difficult to differentiate between regular and surprising periodic events. The notion of a surprising or unusual pattern takes into the likelihood of pattern occurrence to classify account the relative frequency of a pattern with patterns of similar length. The algorithm also takes into account the coverage area of the pattern and it as an outlier pattern. This definition is not limited to the assumption that patterns involving less frequent events are unusual patterns, as described in [19], [20], and [21] nor it requires the training/testing phase, as described in [22].

IV. Conclusion

With this definition, it can also identify outlier patterns that mayinvolve some (or all) frequent events, as itcheck the repetitions of combination of events and not just the individual events. The experimental results show that the proposed algorithm consistently outperforms the existing approach InfoMiner. Additionally, our outlier detection algorithm, being an extension of the ESA periodicity detection framework [23]

It also shown this ESA is not only time efficient but also space efficient [24]. Finally, it are currently working on the following aspect. The definition of surprising patterns can be further improved; some possibilities include the exclusion of user-specified minimum surprise value and integration of standard deviation in the definition of candidate outlier patterns. With this, the candidate outlier patterns might be defined as those which have less than (mean – 2*stdev) repetitions.

REFERENCES

[1]Chandola, V., Banerjee, et.al (2007). Outlier detection: a survey: Technical Report. University of Minnesota, USA

[2] H. Cao, N. Mamoulis, D.W. Cheung, "Discovery of periodic patterns in spatiotemporal sequences",IEEE Trans. Knowl. Data.Eng., 19(4), 2007.

[3] H. Cao, N. Mamoulis, D.W. Cheung, "Discovery of periodic patterns in spatiotemporal sequences",IEEE Trans. Knowl. Data.Eng., 19(4), 2007.

[4] N. Mamoulis , H. Cao , G. Kollios , M.Hadjieleftheriou, Y. Tao , D.W. Cheung, " Mining,indexing, and querying historical spatiotemporaldata" In Proc. 10th ACM SIGKDD Int'l conference on Knowledge discovery and data mining,2004, pp.236-245.

[5] Z. Li, B. Ding, J. Han, R. Kays, P. Nye, "Miningperiodic behaviors for Moving Objects", In. Proc.16th ACM SIGKDD Int'l conference onKnowledge discovery and data mining, 2010.

[6] M. Vlachos, P. S. Yu, and V. Castelli, "On periodicity detection and structural periodic similarity", In proc. of SIAM Int'l Conference on Data Mining, 2005

[7] Z. Li, J.Han, B.Ding, and R. Kays, "Mining periodic behaviors of object movements for animal and biological sustainability studies", Journal of Data Mining and Knowledge Discovery, 2011.

[8]Z.Li, J. Wang, J. Han, "Mining Event Periodicityfrom Incomplete Observations", Proc. 18th ACMSIGKDD Int'l. Conf. Knowledge Discovery and Data Mining, 2012, pp. 444-452.

[9] F. Rasheed, M. Alshalalfa, and R. Alhajj, "Efficient periodicity mining intime series databases using suffix trees," IEEE Trans. Knowl. Data Eng., vol. 23, no. 1, pp. 79–94, Jan. 2011.

[10] D. Gusfield, Algorithms on Strings, Trees, and Sequences. Cambridge, U.K.: Cambridge Univ. Press, 1997.

[11] R. Kolpakov and G. Kucherov, "Finding maximal repetitions in a word in linear time," in Proc. Annu.Symp. Found. Comput. Sci., 1999, pp. 596–604.

[12] E. Ukkonen, "Online construction of suffix trees," Algorithmica, vol. 14,no. 3, pp. 249–260, 1995.

[13] J. Fayolle and M. D. Ward, "Analysis of the average depth in a suffix tree under a Markov model," in Proc. Int. Conf. Anal. Algorithm, Discr.Math.Theor.Comput. Sci., 2005, pp. 95–104.

[14] Y. A. Reznik, "On tries, suffix trees, and universal variable-length-to-block codes," in Proc. IEEE Int. Symp. Inf. Theory, 2002, p. 123.

[15] C.-F. Cheung, J. X. Yu, and H. Lu, "Constructing suffix tree for gigabyte sequences with megabyte memory," IEEE Trans. Knowl. Data Eng.,vol. 17, no. 1, pp. 90–105, Jan. 2005

[16] N. Kumar, N. Lolla, E. Keogh, S. Lonardi, C. A. Ratanamahatana, and L. Wei, "Time-series bitmaps: A practical visualization tool for workingwith large time series databases," in Proc. SIAM Int. Conf. Data Mining,Newport Beach, CA, USA, 2005, pp. 531–535

[17] E. Hunt, R. W. Irving, and M. P. Atkinson, "Persistent suffix trees and suffix binary search trees as DNA sequence indexes," Dept. Comput. Sci.,Univ. Glasgow, Glasgow, U.K., Tech. Rep. TR2000-63, 2000

[18] J. Yang, W. Wang, and P. S. Yu, "Infominer: Mining surprising periodic patterns," in Proc. 7th ACM SIGKDD Int. Conf. Knowl. Discov. Data Mining, 2001, pp. 395–400.

[19]J.Yang,W.Wang, and P.Yu, "InfoMiner+: Mining partial periodic patterns with gap penalties," in Proc. IEEE Int. Conf. Data Mining, Dec. 2002.

[20] J. Yang, W. Wang, and P. S. Yu, "STAMP: On discovery of statistically important pattern repeats in long sequential data," in Proc. SIAM Int. Conf.Data Mining, 2003, pp. 224–238.

[21] E. Keogh, S. Lonardi, and B. Y.-C.Chiu, "Finding surprising patterns in a time series database in linear time and space," in Proc. 8th ACM SIGKDD Int. Conf. Knowl.Discov. Data Mining, 2002, pp. 550–556.

[22] F. Rasheed, M. Alshalalfa, and R. Alhajj, "Efficient periodicity mining in time series databases using suffix trees," IEEE Trans. Knowl. Data Eng., vol. 23, no. 1, pp. 79–94, Jan. 2011