OFFLINE –SUFFIX TREE BASED COMBINATION FORMATION IN PERIODICITY MINING

V. PREMA, FINAL M.E., C.S.E., KARPAGAM UNIVERSITY

E-MAIL:ID:prema.kucs@gmail.com, PH.NO:9994434674

Mr. S. Shahul Hameed, Assistant prof.[Dept of CSE] KARPAGAM UNIVERSITY

Abstract—Periodic pattern mining or periodicity detection has a number of applications, such as prediction, forecasting, detection of unusual activities, etc. The problem is not trivial because the data to be analyzed are mostly noisy and different periodicity types (namely symbol, sequence, and segment) are to be investigated. Accordingly, we argue that there is a need for a comprehensive approach capable of analyzing the whole time series or in a subsection of it to effectively handle different types of noise (to a certain degree) and at the same time is able to detect different types of periodic patterns; combining these under one umbrella is by itself a challenge. In this paper, we present an algorithm which can detect symbol, sequence (partial), and segment (full cycle) periodicity in time series. The algorithm uses suffix tree as the underlying data structure; this allows us to design the algorithm such that its worstcase complexity is $O(k:n^2)$, where k is the maximum length of periodic pattern and n is the length of the analyzed portion (whole or subsection) of the time series. The algorithm is noise resilient; it has been successfully demonstrated to work with replacement, insertion, deletion, or a mixture of these types of noise. We have tested the proposed algorithm on both synthetic and real data from different domains, including protein sequences. The conducted comparative study demonstrate the applicability and effectiveness of the proposed algorithm; it is generally more time-efficient and noise-resilient than existing algorithms.

INTRODUCTION

A time series is a collection of data values gathered generally at uniform interval of time to reflect certain behavior of an entity. Real life has several examples of time series such as weather conditions of a particular location, spending patterns, stock growth, transactions in a superstore, network delays, power consumption, computer network fault analysis and security breach detection, earthquake prediction, gene expression data analysis etc. A time series is mostly characterized by being composed of repeating cycles. For instance, there is a traffic jam twice a day when the schools are open; number of transactions in a superstore is high at certain periods during the day, certain days during the week, and so on. Identifying repeating (periodic) patterns could reveal important observations about the behavior and future trends of the case represented by the time series , and hence would lead to more effective

decision making. In other words, periodicity detection is a process for finding temporal regularities within the time series, and the goal of analyzing a time series is to find whether and how frequent a periodic pattern (full or partial) is repeated within the series.

In general, three types of periodic patterns can be detected in a time series: 1) symbol periodicity, 2) sequence periodicity or partial periodic patterns, and 3) segment or full-cycle periodicity. A time series is said to have symbol periodicity if at least one symbol is repeated periodically. For example, in time series T = abd acb aba abc, symbol a is periodic with periodicity p = 3, starting at position zero (stPos = 0). Similarly, a pattern consisting of more than one symbol maybe periodic in a time series; and this leads to partial periodic patterns. For instance, in time series T = bbaa abbd abca abbc abcd, the sequence ab is periodic with periodicity p = 4, starting at position 4 (stP os = 4); and the partial periodic pattern ab * * exists in T, where * denotes any symbol or don't care mark. Finally, if the whole time series can be mostly represented as a repetition of a pattern or segment, then this type of periodicity is called segment or full-cycle periodicity. For instance, the time series T = abcab abcab abcab has segment periodicity of 5 (p = 5) starting at the first position (stP os = 0), i.e., T consists of only three occurrences of the segment abcab.

In this project URL Reputation data set is used to change the value into range. Data set having three attributes URL, benign and malicious. Various steps are involved in the data translation there are Data selection, data cleaning, data preprocessing, data mining, visualization and interpretation. Data selection is used to select particular data in the data warehouse to get a data is called data mart or data selection. Data cleaning is used to get the data without nose or impurities to remove the noise and impurities is called data mining. Removal of null values, empty values, and missing values. De-duplication is the process of removing old values is known as de-duplication. Data preprocessing is used to change the range of the single data vice versa. Because the algorithm is based on some certain conditions. Algorithm steps is called data mining. Data visualization is nothing but the view of the data. In URL reputation data set malicious data are not suitable for viewing. Benign the data are the acceptable form. Using this URL reputation data set we are collect only the acceptable data set. It will collect these type of data in the time series data set is used to collect the benign data and it stored in the database storage structure layer

.

PROJECT OVERVIEW:

IMPACT ANALYSIS

Before forming the data mining tools it is essential to understand in concept of relation data

base and data sets. Understanding the data set alone will not helping solving these issues any two find out the reason of why the system needs, these process is termed as impact analysis impact analysis placed an important role formation of data mining concepts. In this the impact of the results this essential. Impact analysis is done on fields and where respects rules.

TIME SERIES FORMATION

A time series is a collection of data values gathered generally at uniform interval of time to reflect certain behavior of an entity. Real life has several examples of time series such as weather conditions of a particular location, spending patterns, stock growth, transactions in a superstore, network delays, power consumption, computer network fault analysis and security breach detection, earthquake prediction, gene expression data analysis etc. A time series is mostly characterized by being composed of repeating cycles. For instance, there is a traffic jam twice a day when the schools are open; number of transactions in a superstore is high at certain periods during the day, certain days during the week, and so on. Identifying repeating (periodic) patterns could reveal important observations about the behavior and future trends of the case represented by the time series and hence would lead to more effective decision making. In other words, periodicity detection is a process for finding temporal regularities within the time series, and the goal of analyzing a time series is to find whether and how frequent a periodic pattern (full or partial) is repeated within the series.

OFFLINE FORMATION

we introducing a database storage structure layer outside to loop the offline data. In Existing system we want to search the data in the time series database means it will collect only the online data. For that we are using layer. We have tested in the proposed algorithm the effectiveness is more time-efficient and noise resilient than existing algorithm and also collect the offline data also.

Occ_Vect

We traverse the tree in bottom-up order to construct what we call occurrence vector for each edge connecting an internal node to its parent. We start with nodes having only leaf nodes as children; each such node passes the values of its children (leaf nodes) to the edge connecting it to its parent node. The values are used by the latter edge to create its occurrence vector (denoted occur vec in the algorithm). The occurrence vector of edge e contains index positions at which the substring from the root to edge e exist in the original string. Second, we consider each node v having a mixture of leaf and nonleaf nodes as children.

The occurrence vector of the edge connecting v to its parent node is constructed by combining the occurrence vector(s) of the edge(s) connecting v to its nonleaf child node(s) and the value(s) coming from its leaf child node(s). Finally, until we reach all direct children of the root, we

recursively consider each node u having only nonleaf children. The occurrence vector of the edge connecting u to its parent node is constructed by combining the occurrence vector(s) of the edge(s) connecting u to its child node(s). Applying this bottom-up traversal process on the suffix tree shown in Fig. 1 will produce the occurrence vectors reported in Fig. 2. The periodicity detection algorithm uses the occurrence vector of each intermediate edge (an edge that leads to a nonleaf node) to check whether the string represented by the edge is periodic. The tree traversal process is implemented using the nonrecursive explicit stack-based algorithm presented in [30], which prevents the program from throwing the stack-overflow-exception.

NOISE REDUCTION

Noise reduction is a very challenging problem. The nature and the characteristics of noise change significantly from application to application. In addition, noise characteristics vary in time. It is therefore very difficult to develop a versatile algorithm that works in diversified environments. Also, the objective of a noise reduction system may depends on the specific context and application. In existing system noise deduction is not done. Because in online we do no consider about the noise in the online. So in the offline noise reduction is done by the use of noise deduction algorithm.

RESULT FINDING

Result finding module is to essential in respect to finding of results in the process of generating the concept of time series prediction. The complete algorithm was analyser and implemented proper visualization technique are to be used in bringing the solution to the understandable formats to the users is essential. The model which given this support done in the system this would helping generating better solution the users required.

ABOUT THE DATASET:

In this project we are using URL repository data set. Data set having three fields they are URL, Malicious, benign. In URL (Uniform resource locator), filed have the mail address and malicious data are not suitable for viewing. Unwanted information are stored in this malicious URL. If we want to search the data in the google means if both the malicious and benign data are displayed so to avoid the malicious data in the database.

BACKGROUND STUDY AND RELATED WORK:

The rest of the paper is organized as follows: Related work is presented in Section 2. The periodicity detection problem in time series is formulated in Section 3. The proposed algorithm is presented in Section 4 along with algorithm analysis and the utilized optimization strategies.

A SUFFIX TREE BASES NOISE RESILIENT ALGORITHM FOR PERIODICITY DETECTION IN TIME SERIES DATABASES

Periodicity detection has been used extensively in predicting the behavior and trends of

time series databases. In this paper, we present a noise resilient algorithm for periodicity detection using suffix trees as an underlying data structure. The algorithm not only calculates symbol and segment periodicity, but also detects the partial (or sequence) periodicity in time series. Most of the existing algorithms fail to perform efficiently in presence of noise; although noise is an inevitable constituent of real world data. The conducted experiments demonstrate that our algorithm performs more efficiently compared to other algorithms in presence of replacement, insertion, deletion or a mixture of any of these types of noise.

PROBLEM DEFINITION:

Mining of data periods this extraction of periods from set of data is known as periodicity mining. We define periodicity mining as the detection of frequent periodic patterns where the periodicity rate (period length) is also unknown. Although one may argue that visual inspection of the time plot of the time series may lead to the identification of potential periodicity rates, huge amounts of data and the evolution nature of time series streams complicate such visual inspection. Moreover, complex applications like stock market do not follow obvious periodicity rates (daily or weekly). A period may span a long rate (e.g, quarterly), which is not obvious to observe visually. Therefore, periodicity mining comprises mainly two steps. The first step, termed *Periodicity Detection*, is to discover potential periodicity rates. The second step, termed *Mining Periodic Patterns*, is to detect the frequent periodic patterns of each discovered periodicity rate.

A time series is a collection of data values gathered generally at uniform interval of time to reflect certain behavior of an entity. Real life has several examples of time series such as weather conditions of a particular location, spending patterns, stock growth, transactions in a superstore, network delays, power consumption, computer network fault analysis and security breach detection, earthquake prediction, gene expression data analysis etc. A time series is mostly characterized by being composed of repeating cycles. For instance, there is a traffic jam twice a day when the schools are open; number of transactions in a superstore is high at certain periods during the day, certain days during the week, and so on. Identifying repeating (periodic) patterns could reveal important observations about the behavior and future trends of the case represented by the time series and hence would lead to more effective decision making. In other words, periodicity detection is a process for finding temporal regularities within the time series, and the goal of analyzing a time series is to find whether and how frequent a periodic pattern (full or partial) is repeated within the series.

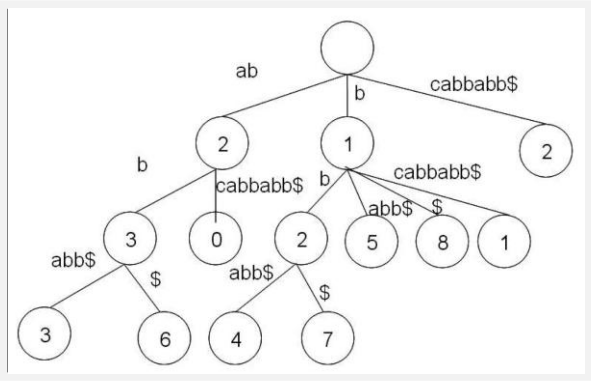

Fig. 4.2.1. The suffix tree for the string abcabbabb$.

CONCLUSION:

In this paper, we have presented STNR as a suffix-tree-based algorithm for periodicity detection in time series data. Our algorithm is noise-resilient and run in $O(k:n^2)$ in the worst case. The single algorithm can find symbol, sequence (partial periodic), and segment (full cycle) periodicity in the time series. It can also find the periodicity within a subsection of the time series. We performed several experiments to show the time behavior, accuracy, and noise resilience characteristics of the data. We run the algorithm on both real and synthetic data. The reported results demonstrated the power of the employed pruning strategies.

The algorithm uses suffix tree as the underlying data structure; this allows us to design the algorithm. We are using an improved algorithm combing both periodicity detection algorithm and noise resilient periodicity detection. And also we introducing and extra layer is database storage structure layer outside to loop the offline data. In Existing system we want to search the data in the time series database means it will collect only the online data. We have tested in the proposed algorithm the effectiveness is more time-efficient and noise resilient than existing algorithm and also collect the offline data also.

Currently, we are working on online periodicity detection where the suffix tree is constructed online, i.e., extended while the algorithm is in operation. This type of stream mining has a large number of applications especially in sensor networks, ubiquitous computing, and DNA sequence mining. We are also implementing the disk-based solution of the suffix tree to extend capabilities beyond the virtual memory into the available disk space. In [6], the occurrences which drift from the expected position within an allowable limit.

REFERENCES:

1. M.G. Elfeky, W.G. Aref, and A.K. Elmagarmid, **"Periodicity Detection in Time Series Databases"**, IEEE Trans. Knowledge and Data Eng., vol. 17, pp. 875-887, July 2005.

2. K. Y. Huang and C.H. Chang, "**SMCA: A General Model for Mining Asynchronous Periodic Patterns in Temporal Databases,"** IEEE Trans. Knowledge and Data Eng., vol. 17, no.6, pp. 774-785, June 2005.

3. F. Rasheed and R. Alhajj, "**Using Suffix Trees for Periodicity Detection in Time Series Databases,"** Proc. IEEE Int'l Conf. Intelligent System, Sept. 2008.

4. A. Al Rawi, A. Lansari, and F. Bouslama, " **A New Non-Recursive Algorithm for Binary Search Tree Traversal,"** Proc. IEEE Int'l Conf. Electronics, Circuits and systems(ICECS), vol. 2,pp.770-773, Dec. 2003