International Journal of Computer Applications (0975 - 8887)
Volume 178 - No.11, May 2019

Mining Recurring Patterns in Time Series

Dharmesh Bhalodiya
Computer Engineering
Atmiya University
Rajkot, Gujarat, India

Jaydeep Tadhani
Computer Engineering
Atmiya University
Rajkot, Gujarat, India

Rajesh Davda
Computer Engineering
Atmiya University
Rajkot, Gujarat, India

ABSTRACT

Periodic pattern mining consists of finding patterns that exhibit either complete or partial cyclic repetitions in a time series. Past studies on partial periodic search focused on finding regular patterns, i.e., patterns exhibiting either complete or partial cyclic repetitions throughout a series. An example regular pattern of Bat, Ball stats that customers have been purchasing items Bat and Ball alost every day throughout the year. The type of partial periodic pattern is recurring patens, i.e., patterns exhibiting cyclic repetitions only for particular time intervals within a series. Its a very difficult task to identify those periodic frequent patterns within given threshold in time. To overcome these problem, we introduced modification in traditional PR-tree structure. And this structure improves overall efficiency by running time, Periodic Frequent Pattern generation and Memory consumptions

Keywords

Recurring Patterns, RP-tree, Time Series

1. INTRODUCTION

Data mining is the combination of different technique like data base, data warehouse, pattern mining, data visualization, signal processing etc. Data mining uses these techniques to extract the knowledge or pattern from the large collection of data.[1]
Time series is a collection of events, obtained from sequential measurement over time. Thus, time series contain sequence values that repeated interval of times. Hence, it results into very large database. We observed that it is tedious to mine periodic pattern in very large database, the reason is that the occurrence behavior of the pattern can differ over a period of time affecting periodically occurring patterns to be periodic or not periodic; users might be interested to find only those patterns that might have appeared almost periodically all over the database.
In real world customers buy seasonal products, for example customer buy a woolen cloths in winter umbrellas and raincoats in monsoon and air conditioners in summer, Thus to decide which products are in high demand is a periodic task, this is called recurring patterns that exhibiting periodic behavior only for particular time intervals with in a series. The purpose of this paper is to discover recurring patterns by addressing mining challenges.
Unfortunately, finding recurring patterns is a non-trivial task because of the following reasons:

(1) Each recurring pattern is associated with time stamp pertaining to its durations of periodic appearances within the data. Obtaining this information is challenging because the information can vary within and across patterns. Most current periodic pattern mining approaches take into account a time series as a symbolic sequence; so, it did not take into account the actual temporal information of events.

(2) Since regular patterns exhibit periodic behavior throughout a series with some exceptions, regular pattern mining algorithms do not obtain temporal information pertaining to the durations of periodic appearances of a pattern within the series. As a result, these algorithms cannot be extended for finding recurring patterns.

(3) In real-life, recurring patterns involving rare items can be interesting to users. For example, the knowledge pertaining to rare events, such as cascading failures, are more important than regular events for a network administrator. However, finding such patterns is difficult since rare items appear infrequently in the data.

In this paper, we propose a model that addresses all the above-mentioned issues while finding recurring patterns. In particular, our model takes into account time series as a time-based sequence and models it as a transactional database with transactions ordered in respect to a particular timestamp (without loss of generality). Our model consists of three novel measures, periodic-support, periodic-interval and recurrence, to determine the dynamic periodic behavior of recurring patterns. Periodic-support determines the number of consecutive cyclic repetitions of a pattern in a subset of data. Periodic-interval determines the time interval (or window) pertaining to the periodic appearances of a pattern within a series. Recurrence determines the number of interesting periodic intervals of a pattern. Finally, we propose a pattern-growth algorithm along with an efficient pruning technique to discover recurring patterns effectively. We call our algorithm recurring pattern-growth (RP-growth). The rest of the paper is organized as follows. Section 2 describes work on mining periodic patterns. Section 3 introduces our approach of recurring patterns. Section 4 presents RP-growth. Sections 5 reports on the experimental results. Finally, Section 6 concludes the paper with future research directions.

2. RELATED WORK

Since the presentation of fractional intermittent examples of periodic pattern [2], the problem of finding these examples has got a lot of attention. They demonstrate utilized as a part of every one of these examinations, in any case, continues as before. That is,

it considers a period arrangement as an emblematic grouping and discovers all examples utilizing the accompanying two stages:

(1) Partition the symbolic sequence into distinct subsets (or period-segments) of a fixed length (or period).
(2) Discover all partial periodic patterns that satisfy the given minimum support (minSup), which controls the minimum number of period-segments that a pattern must cover though the sequence.

In the research done by Syan Tanbeer [3] discovered a novel concept for mining in transactional databases.A major limitation of the above studies is that did not take into account the actual temporal information of the events within a sequence.
In visit pattern mining, minSup controls the base number of appearances of an example all through the information. Notwithstanding, in halfway occasional design mining, minSup controls the base number of occasional appearances (or cyclic repetitions) of an example all through the information. Therefore, the halfway occasional examples found in all the above examinations [5] [6] [7] [8] [9] leads to normal examples. To address this issue, Ma and Hellerstein [4] modeled a time series as a time-based sequence and proposed a model to discover a class of partial periodic patterns known as p-patterns. In this model, an any pattern is considered partial periodic if its total number of periodic appearances throughout the sequence satisfies the given minSup. It should be noted that the concept of minSup is not the same in both frequent pattern mining and partial periodic pattern mining. In frequent pattern mining, minSup controls the minimum number of appearances of a pattern throughout the data. Our study, on the other hand, was focused on discovering recurring patterns in a time-based sequence. Moreover, Ma and Hellersteins model cannot be extended for finding recurring patterns. The reasons are as follows:

(1) Their model fails to obtain the temporal information pertaining to the durations of periodic appearances of a pattern within the data.
(2) Finding p-patterns with a single minSup leads to the dilemma known as the rare item problem [10].

If minSup is set too high, those patterns that involve rare items not be found. To find patterns involving both frequent and rare items, minSup must be set very low. However, this can lead to combinatorial explosion producing too many patterns. In particular, many uninteresting aperiodic patterns can be discovered as partial periodic patterns. For example, if we set a low minSup, say minSup = 5%, then we will be discovering an uninteresting aperiodic pattern that has only 5% of its periodically appearances throughout the data as a partial periodic pattern.
Manziba and Farhan[11] introduced a new method of periodicity mining in time series database to generate patterns by addming more flexibilities and choices to user. In their approach they provided better user interaction to user for providing periodic values and number of intermediate events. R.Uday Kiran introduced a novel method based on frequent pattern. Related work done in the above research [12] stated FPPM can measure variable length flexible periodic patterns by neglecting unimportant or undesired events.

3. PROPOSED METHORD

In this section, we first introduce existing pruning technique to reduce the computational cost of finding patterns.
The space of items in a dataset gives rise to a generation of subset lattice. An item-set lattice is a conceptualization of search space

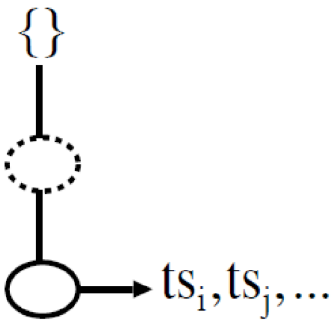


Fig. 1. Traditional RP-Tree structure.

while finding user-interest patterns. The anti-monotonic named property helpful for reduce the search space [13]. Unfortunately, recurring patterns never satisfy this property. In impact of that search space, which in turn increases the computational cost of new finding recurring patterns.

$$E_{rec}(X) = \sum_{i=1}^{|ps^x|}(\frac{ps_i^x}{minps}) \qquad if E_{rec}(X) < minRec$$

The $Erec(X)$ signifies the upper bound of recurrence that a superset of X can have in the database. Thus, we call $Erec(X)$ will be estimated maximum recurrence of a superset of X.
To address this issue, RP-growth introduced an alternative tree structure known as an Recurring Pattern-tree (RP-tree) [12]. RP-growth algorithm involved the following two steps:

(1) construction of an RP-tree and
(2) recursive mining of the RP-tree to discover the complete set of recurring patterns

Authors of RP-growth also introduced improved structure of Tree as shown below. Only Last item of pattern contain transaction list in RP-Tree structure.

3.1 An Idea

We introduce modification in RP-tree with size indication of transaction list includes a prefix-tree and a candidate item list, called the RP-list. The RP-list consists of each distinct item with support, estimated maximum recurrence $(Erec)$, and a pointer pointing to the start node in the prefix-tree carrying the item.
RP-tree explicitly maintain the occurrence information for each transaction by keeping an occurrence timestamps list, called a ts-list. To achieve memory efficiency, only the last node of each transaction maintains the ts-list which is already given in existing method, and we introduce new parameter size count of transaction list. The conceptual structure of an extended RP-tree is shown in figure 3.1. This extended tree contains one extra size parameter, else structure same as existing approach. The size parameter stores count of transaction list; it will utilize when further scanning of RP-tree.
Old method maintains list called $subDB$, which store pair of first node $(startTS)$ and last node $(endTS)$ from transaction list (ts) for each node. It is used to get recurrence of pattern $(getRecurrence)$. This process repeated every time whenever recurrence count wanted. To remove this bottleneck, we introduce change in RPtree structure. Algorithm to solve above mentioned issue represented in ALGORITHM 1
As we observe from above ALGORITHM 1, it reduces iterations for traversal of $tslist$ of given pattern.

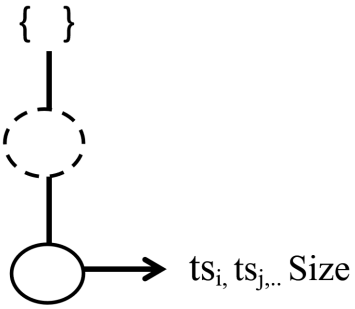


Fig. 2. Modified structure of RP-Tree

ALGORITHM 1 getTScount

1: Let *count* be a variable that records occurrence count of patt and *minRec*
2: return ($TS^x.Size() \geq minRec)?\, true: false$);

Table 1.
Dataset

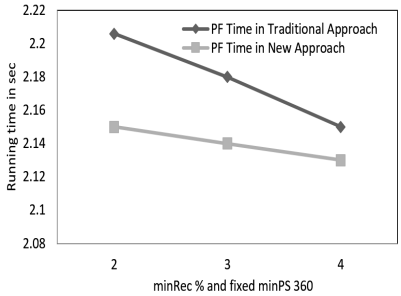
Dataset Name	# of Transactions	# of Items	Kind
T10I4D100K	100000	941	Synthetic
Retail	88162	16470	Real world

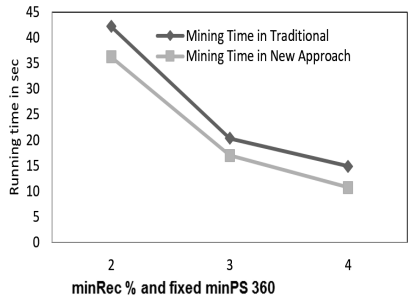
4. EXPERIMENTS

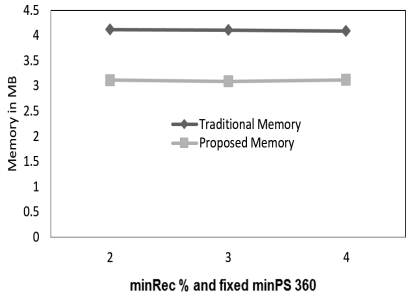
In this section, we evaluate proposed modified RPtree structure in traditional RP growth, codebase was given by author for extending method. The base algorithm written in GNU C++ of Ubuntu 16.04 with 16 GB of primary memory. as author of base paper already declared that time-based sequences dataset not available publically. Datasets that are used while in experiments mentioned below. We take two different kind synthetic and real world to measure correctness of proposed method.

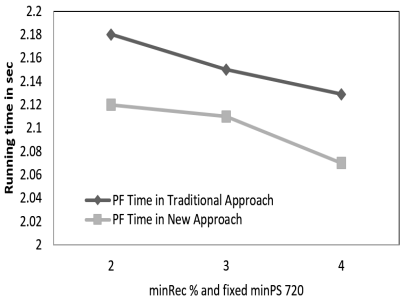
4.1 Results and discussions

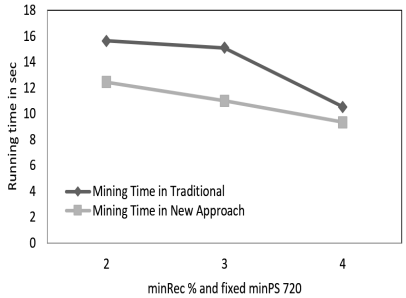
All the results presented in this papers are average of three consecutive readings. We considered three different $minPS$ values for our experiments those are 360, 720 and 1440 respectively. Experimental result dispute runtime($mining$), Periodic Frequent pattern generation from RP-tree(PF) and Memory consumption (MB) with traditional method. Below graphs shows the mining and PF time as well as MBs required by RP-growth vs Proposed while mining the recurring patterns in databases. The changes in the per, $minPS$ and $minRec$ threshold values shows a different eects on runtime consumption as in the generation of recurring patterns. The proposed algorithm exposed the complete set of recurring patterns at a judicious runtime even at low $minPS$ thresholds.

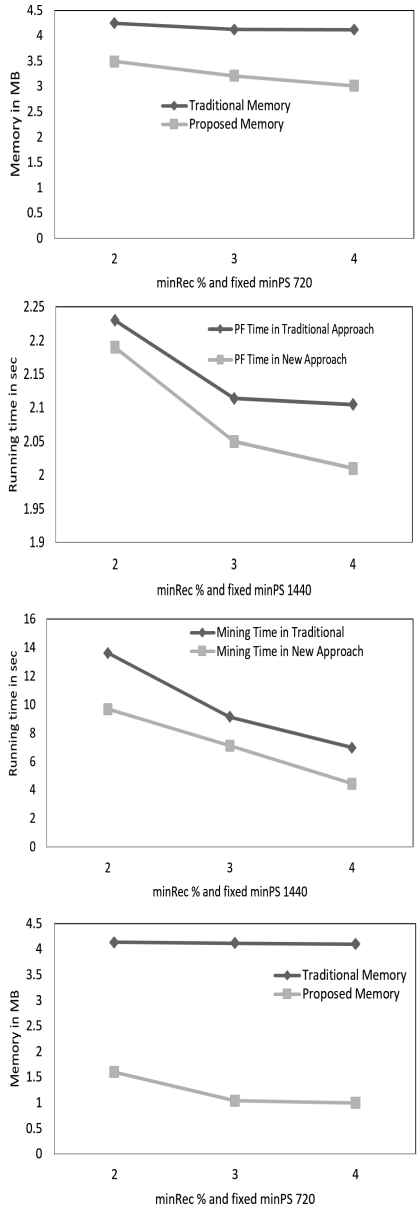












5. CONCLUSIONS

We proposed a new extension of partial periodic patterns known as recurring patterns and discussed the usefulness of these patterns in various real-world applications. The patterns discovered with the proposed model do not satisfy the anti-monotonic property. Therefore, we proposed a modified tree structure for pruning technique to reduce the overall cost of mining these patterns. We also proposed an algorithm to discover the recurring patterns effectively. Experimental results dispute the algorithm is able to find rare items problem as previous method and so the algorithm is efficient. The usefulness of the recurring patterns was discussed by comparing them against the periodic-frequent and p-patterns

6. REFERENCES

[1] Jiawei Han and Micheline Kamber, Data Mining Concepts and Techniques, Second Edition, Morgan Kaufmann Publications, 2006.

[2] Surana A., Kiran R.U., Reddy P.K. An Efficient Approach to Mine Periodic-Frequent Patterns in Transactional Databases. In: Cao L., Huang J.Z., Bailey J., Koh Y.S., Luo J. (eds) New Frontiers in Applied Data Mining. PAKDD 2011. Lecture Notes in Computer Science, vol 7104. Springer, Berlin, Heidelberg 2012 doi:10.1007/978-3-642-28320-8_22

[3] Syed Khairuzzaman Tanbeer, Chowdhury Farhan Ahmed, Byeong-Soo Jeong, and Young-Koo Lee, Discovering Periodic-Frequent Patterns in Transactional Databases, Springer-Verlag Berlin Heidelberg, pp. 242253, 2009.

[4] S. Ma and J. Hellerstein, Mining partially periodicevent patterns with unknown periods, in ICDE, 2001, pp. 205214.

[5] J. Han, G. Dong, and Y. Yin, Efficient mining of partial periodic patterns in time series database, in ICDE, 1999, pp. 106115.

[6] R. Yang, W. Wang, and P. Yu, Infominer+: mining partial periodic patterns with gap penalties, in ICDM, 2002, pp. 725728.

[7] Berberidis C., Vlahavas I., Aref W.G., Atallah M., Elmagarmid A.K. (2002) On the Discovery of Weak Periodicities in Large Time Series. In: Elomaa T., Mannila H., Toivonen H. (eds) Principles of Data Mining and Knowledge Discovery. PKDD 2002. Lecture Notes in Computer Science (Lecture Notes in Artificial Intelligence), vol 2431. Springer, Berlin, Heidelberg

[8] H. Cao, D. Cheung, and N. Mamoulis, Discovering partial periodic patterns in discrete data sequences, in Advances in Knowledge Discovery and Data Mining, 2004, vol. 3056, pp. 653658.

[9] W. G. Aref, M. G.s Elfeky and A. K. Elmagarmid, "Incremental, online, and merge mining of partial periodic patterns in time-series databases," in IEEE Transactions on Knowledge and Data Engineering, vol. 16, no. 3, pp. 332-342, Mar 2004. doi: 10.1109/TKDE.2003.1262186

[10] Bing Liu, Wynne Hsu, and Yiming Ma. 1999. Mining association rules with multiple minimum supports. In Proceedings of the fifth ACM SIGKDD international conference on Knowledge discovery and data mining (KDD '99). ACM, New York, NY, USA, 337-341. DOI: http://dx.doi.org/10.1145/312129.312274

[11] Manziba Akanda Nishi, Chowdhury Farhan Ahmed, Md. Samiullah, and Byeong-Soo Jeong. 2013. Effective periodic pattern mining in time series databases. Expert Syst. Appl. 40, 8 (June 2013), 3015-3027. DOI: https://doi.org/10.1016/j.eswa.2012.12.017

[12] R. Uday Kiran, Haichuan Shang, Masashi Toyoda and Masaru Kitsuregawa, Discovering Recurring Patterns in Time Series, Proc.18thInternational Conference on Extending Database Technology (EDBT), March 23-27, 2015

[13] R. Agrawal, T. Imielinski, and A. Swami, Mining association rules between sets of items in large databases, in SIGMOD, 1993, pp. 207216.