Discovering Rare Correlated Periodic Patterns in Multiple Sequences

Philippe Fournier-Viger[a,*], Peng Yang[b], Zhitian Li[b], Jerry Chun-Wei Lin[c], Rage Uday Kiran[d]

[a]*School of Humanities and Social Sciences, Harbin Institute of Technology (Shenzhen), China*
[b]*School of Computer Science and Technology, Harbin Institute of Technology (Shenzhen), China*
[c]*Department of Computing, Mathematics, and Physics, Western Norway University of Applied Sciences (HVL), Bergen, Norway*
[d]*National Institute of Information and Communication Technologies, Tokyo, Japan*

Abstract

Periodic-Frequent Pattern Mining (PFPM) is an emerging problem, which consists of identifying frequent patterns that periodically occur over time in a sequence of events. Though PFPM is useful in many domains, traditional algorithms have two important limitations. First, they are not designed to find rare patterns. But discovering rare patterns is useful in many domains (e.g. to study rare diseases). Second, traditional PFPM algorithms are generally designed to find patterns in a single sequence, but identifying periodic patterns that are common to a set of sequences is often desirable (e.g. to find patterns common to several hospital patients or customers). To address these limitations, this paper proposes to discover a novel type of patterns in multiple sequences called Rare Correlated Periodic Patterns. Properties of the problem are studied, and an efficient algorithm named MRCPPS (Mining Rare Correlated Periodic Patterns common to multiple Sequences) is presented to efficiently find these patterns. It relies on a novel RCPPS-list structure to avoid repeatedly scanning the database. Experiments have been done on several real datasets, and it was observed that the proposed MRCPPS algorithm can efficiently discover all rare correlated periodic patterns common to multiple sequences, and filter many non rare and correlated patterns.

Keywords: Periodic pattern, Rare pattern, Correlated pattern, Sequences

1. Introduction

Frequent Pattern Mining (FPM) [1, 2, 3] is the task of discovering patterns in a database that have an occurrence frequency (support) that is no less than a *minimum support threshold* (*minsup*), set by the user. FPM has been applied in several domains [4, 5]. But most studies have focused on discovering patterns such as frequent itemsets and association rules, and do not consider the time or sequential ordering between symbols (e.g. events). There are many applications for which insightful information can be revealed by analyzing when a pattern occurs. An emerging data mining task of this type is Periodic-Frequent Pattern Mining (PFPM) [6, 7, 8, 9, 10, 11, 12, 13, 14, 15], which is a generalization of FPM. PFPM consists of identifying frequent patterns that periodically occur over time. An example of periodic pattern in market basket data is that a customer buys some items every week. Finding such patterns can help understanding customer behavior, and thus supports the design of appropriate marketing strategies. PFPM has numerous applications, including website user behavior analysis [16], genetic data analysis [17], cross-marketing in retail stores [12], and mobility intention analysis [18]. Even though PFPM is useful in various domains, traditional algorithms have two important limitations. First, most algorithms for identifying periodic patterns focus on discovering frequent patterns, but periodic patterns that are rare are also interesting. For example, rare periodic patterns could be found to study rare diseases in medical data (symptoms caused by latent viruses

*Corresponding author
Email addresses: philfv@hit.edu.cn (Philippe Fournier-Viger), pengyeung@163.com (Peng Yang), tymonlee1@163.com (Zhitian Li), jerrylin@ieee.org (Jerry Chun-Wei Lin), uday.rage@gmail.com (Rage Uday Kiran)

that appear periodically). The second limitation is that algorithms for discovering periodic patterns are generally designed to find patterns in a single sequence, but identifying periodic patterns that are common to a set of sequences is also desirable. For instance, in market basket analysis, it is desirable to discover periodic patterns that are common to several customers (e.g. many customers periodically buy milk and bread) to understand their behavior.

Addressing these two limitations is not straightforward. To cope with multiple sequences, a naive solution is to apply a traditional PFPM algorithm [6] on each sequence and then to combine the patterns found in each sequence to find periodic patterns common to multiple sequences. But this is inefficient as a huge number of patterns would have to be kept in memory, and combining these patterns would be very costly. Recently, an algorithm named PHUSPM [14] has been proposed to mine periodic patterns when considering multiple sequences as a sequence. But this algorithm does not guarantee that patterns are periodic inside each sequence. To find patterns that are truly periodic in many sequences, we have recently proposed the MPFPS algorithm [15]. However, how to extend that algorithm to mine rare periodic patterns is not trivial.

A simple solution to mine rare periodic patterns in a single sequence or multiple sequences is to adapt algorithms to replace the *minimum support threshold* by a *maximum support threshold* ($maxSup$), and then find patterns having a support lower than that threshold. However, this solution does not allow to reduce the search space because subsets of a rare pattern can be frequent patterns. Hence, all rare and frequent patterns (having a support greater than $maxSup$) would have to be checked to find the rare patterns. Thus, novel strategies must be designed to reduce the search space and efficiently mine rare periodic patterns. Besides, a challenge is that a huge number of rare periodic patterns can be spurious due to their low occurrence frequencies. Thus, an appropriate technique must be used to filter spurious patterns.

Moreover, to mine rare periodic patterns, the concept of periodic patterns must also be revised. In traditional PFPM, a pattern is deemed periodic if all its periods (number of transactions between consecutive occurrences) are no larger than a user-specified *maximum periodicity threshold* ($maxPer$). But applying this definition to mine rare periodic patterns would pose a problem because rare patterns generally have very large periods (see Lemma 1).

In this paper, we address the above challenges by proposing a novel framework for mining Rare Correlated Periodic Patterns common to multiple Sequences (RCPPSs). The major contributions of this work are summarized as follows:

- To find rare periodic patterns that are not spurious, this paper proposes to use the *bond* correlation measure to identify strongly correlated periodic patterns. By combining this measure with the concept of periodic patterns, a new type of patterns is defined called Rare Correlated Periodic Patterns. Effective pruning properties are implemented to discover these patterns in multiple sequences.
- To evaluate if rare patterns are periodic (appear regularly) in each sequence, a new measure is defined, which is the standard deviation of periods. Moreover, a novel periodicity measure named *sequence periodic ratio* (ra) is defined to find patterns that are periodic in multiple sequences. To effectively reduce the search space, an *upper-bound* on the ra measure, called **upBondRa**, is developed and a novel pruning property is proposed.
- An algorithm named MRCPPS (Mining Rare Correlated Periodic Patterns common to multiple Sequences) is proposed to efficiently find all rare correlated periodic patterns. It relies on a novel RCPPS-list structure to avoid repeatedly scanning the database[1].
- Experiments have been done on several real datasets. It is observed that the proposed MRCPPS algorithm can efficiently discover all rare correlated periodic patterns common to multiple sequences, and that interesting patterns are revealed.

The rest of this paper is organized as follows. Section 2 reviews related work. Section 3 defines the problem of RCPPS. Section 4 presents the MRCPPS algorithm. Section 5 describes the experimental evaluation. Finally, Section 6 draws the conclusion and discusses future work.

[1]Note that MRCPPS extends the MPFPS algorithm [15] to mine rare correlated periodic patterns. This extension is done for the special issue of the best papers of the DAWAK 2018 conference, where MPFPS was published.

2. Related work

The following paragraphs survey related work on periodic-frequent pattern mining and rare correlated pattern mining.

2.1. Periodic Frequent Pattern Mining

The task of Frequent Itemset Mining (FIM) was first proposed to discover the sets of items (called frequent itemsets) that appear in numerous customer transactions [4, 5]. But it can be more generally viewed as the task of discovering frequently co-occurring values in records of a binary attribute database. To discover frequent itemsets, the Apriori [2] algorithm was first defined. Apriori first scans the database to find frequent items. Then, Apriori recursively joins small patterns to produce patterns containing more items. Apriori scans the database to evaluate the frequency of each pattern and outputs the frequent itemsets. Apriori is a correct and complete algorithm. However, it performs multiple database scans that can be very time-consuming. The AprioriTID [2] and Eclat [3] algorithms address this issue by scanning a database once to create a structure called *tid-list* that stores the list of transactions where each item appears. Then, the *tid-list* of larger patterns are found by joining the tid-lists of some of their sub-patterns. The tid-list structure is useful as it allows to efficiently calculate the support of a pattern without scanning the database. The key difference between AprioriTID and Eclat is that the former utilizes a breadth-first search while the latter apply a depth-first search based on the concept of *equivalence classes* [3]. Though, FIM algorithms have been used in many domains and several other algorithms have been developed thereafter [4, 5], FIM is inappropriate for identifying patterns that appear periodically in a database.

To identify periodic patterns, Tanbeer et al. [6] defined the task of PFPM in a sequence of transactions (events). They found that interesting patterns regularly appear in data, and proposed a pattern-growth algorithm [6] to identify these patterns. In that work, an itemset is said to be periodic in a sequence if it appears a minimum number of times and the number of transactions between each occurrence is not greater than a *minPer* (minimum periodicity) threshold. Then, several more efficient algorithms have been designed, and variation of the problem of mining periodic patterns in a single sequence have been proposed [7, 8, 9, 10, 11, 12, 13, 14, 19]. For instance, the MTKPP algorithm was introduced to find the top-k most frequent periodic patterns in a sequence [7]. Rashid et al. [9] proposed an approach to mine frequent patterns that occur at regular intervals in a transactional database. In that approach, periodicity is measured based on the variance of the time between pattern occurrences. In another work, Kiran et al. [19] addressed the "rare item problem" by proposing to define a minimum support threshold for each item rather than using the same threshold for all items. To find patterns that are periodically bought by customers and generate a high profit, two algorithms named PHM[12] and PHUSPM[14] were proposed that consider the utility of patterns. The utility is a numeric measure of importance such as profit [20, 21]. To find stable periodic patterns in a transaction database with timestamps, the SPP-Growth algorithm was recently proposed [22].

Though there are many papers on periodic pattern mining, most studies focus on discovering frequent patterns in a single sequence. The topic of mining rare correlated periodic patterns in multiple sequences has not been addressed. The next subsection reviews related work about rare and correlated pattern mining.

2.2. Rare Correlated Pattern Mining

The goal of FIM and association rule mining is to identify interesting relationships between values (items) in a database [2, 23, 24]. Two major types of patterns can be found: frequent and rare patterns. In recent years, there has been an increasing focus on discovering rare patterns because it has numerous high impact applications such as detecing computer attacks and fraudulent credit card transactions [25, 26]. However, rare pattern mining is challenging as an enormous number of candidates may be generated to mine rare patterns because the support measure cannot be used to reduce the search space as in traditional FIM. Moreover, a challenge is that some rare patterns may be spurious due to their low occurrence frequency. To reduce the computational complexity of mining rare patterns and avoid finding spurious patterns, recent studies have considered mining rare correlated patterns [26, 27, 28, 29]. To measure correlation, various measures have been used in FIM such as the any-confidence and all-confidence [13, 24, 30], frequence

affinity [31], coherence [32], and bond [24, 29, 30, 33, 34]. Although the above studies can find useful patterns in many applications, they do not consider the periodical behavior of patterns. To our knowledge, only Kiran et al. [13] has employed a measure of correlation in PFPM (the *all-confidence* measure). But it is not for discovering rare patterns and it is only for mining patterns in a single sequence. In this paper, we integrate the *bond* measure to propose a general model and algorithm for mining rare correlated periodic patterns in multiple sequences. The *bond* measure was selected because it is a simple and effective correlation measure, and it has recently attracted the attention of many researchers [24, 29, 30, 33, 34].

3. Definitions and problem statement

This section first presents the problem of discovering rare correlated periodic patterns in a single sequence using a novel measure called the periodic standard deviation and the bond measure to filter uninteresting periodic patterns. Then, the problem of rare correlated periodic pattern mining is generalized to mine patterns in multiple sequences (a sequence database).

Definition 1 (Sequence database). Let I be a set of items (symbols or binary values) occurring in a database. A subset $X \subseteq I$ is called an itemset. The length of an itemset containing k items is k. Moreover, such itemset is called a k-itemset. A sequence s is an ordered list of itemsets $s = \langle T_1, T_2, \cdots, T_m \rangle$, such that $T_j \subseteq I$ $(1 \leq j \leq m)$, T_j is called a transaction, and j is called the TID (Transaction IDentifier) of T_j. A sequence database SDB is an ordered set of n sequences, denoted as $SDB = \langle s_1, s_2, \cdots, s_n \rangle$. The sequence s_i of SDB $(1 \leq i \leq n)$ is said to be the i-th sequence of SDB, and its Sequence IDentifier (SID) is said to be i. An itemset X is said to appear in a sequence $s_a = \langle A_1, A_2, \cdots, A_k \rangle$, denoted as $X \sqsubseteq s_a$, iff there exists an integer $1 \leq q \leq k$ such that $X \subseteq A_q$. In such case X is also said to occur in transaction A_q. And more generally, a sequence $s_a = \langle A_1, A_2, \cdots, A_k \rangle$ is said to be *contained* in another sequence $s_b = \langle B_1, B_2, \cdots, B_l \rangle$, denoted as $s_a \sqsubseteq s_b$, iff there exist integers $1 \leq q1 \leq q2 \leq \cdots \leq qk \leq l$ such that $A_1 \subseteq B_{q1}, A_2 \subseteq B_{q2}, \cdots, A_k \subseteq B_{qk}$.

Table 1: A sequence database

SID	Sequence
1	$\langle \{a, c, e\}, \{a, b, e\}, \{a, d\}, \{a, b, e\}, \{a, c\} \rangle$
2	$\langle \{c\}, \{a, b, c, e\}, \{c, d\}, \{a, b, c, e\}, \{a, b, d\} \rangle$
3	$\langle \{b, c\}, \{a, b\}, \{a, c, d\}, \{a, c\}, \{a, b\} \rangle$
4	$\langle \{a, b, d, e\}, \{a, b\}, \{a, c\}, \{a, b, d, e\}, \{a, d\} \rangle$

In the following, the sequence database shown in Table 1 will be used as running example. It contains four sequences. The third sequence (s_3) contains five itemsets. The first itemset contains two items (b and c). Hence, it is a 2-itemset. The itemset $\{a, c\}$ appears in the third and fourth transactions of sequence s_3. The sequence $\langle \{b\}, \{a, c\} \rangle$ also appears in s_3. To identify periodic patterns in a sequence, the concept of period was proposed in PFPM [6].

Definition 2 (Periods of an itemset in a sequence). Consider a sequence s_i of a database SDB and an itemset X. The ordered list of transactions where X appears in s_i is denoted as $TR(X, s_i) = \langle T_{g(1)}, T_{g(2)}, \cdots, T_{g(k)} \rangle \sqsubseteq s_i$. Moreover, k is called the support (number of appearances) of X in s_i and is denoted as $sup(X, s_i) = |TR(X, s_i)|$. Let $T_{g(z)}$ and $T_{g(z+1)}$, $z \in [1, k-1]$ be a pair of consecutive transactions where X appears in s_i. The period of two consecutive transactions $T_{g(z)}$ and $T_{g(z+1)}$ for itemset X is $per(T_{g(z)}, T_{g(z+1)}) = g(z+1) - g(z)$. The periods of X in a sequence s_i are $pr(X, s_i) = \{per_1, per_2, \cdots, per_{k+1}\}$ where $per_1 = g(1) - g(0), per_2 = g(2) - g(1), \cdots, per_{k+1} = g(k+1) - g(k)$, where $g(1), g(2), \cdots, g(k)$ are the TIDs of transactions where itemset X occurs and $g(0)$ and $g(k+1)$ are defined as $g(0) = 0$ and $g(k+1) = |s_i|$, where $|s_i|$ is the length of s_i.

For example, the itemset $\{a, e\}$ occurs in transaction T_1, T_2 and T_4 of sequence s_1, i.e. $TR(\{a, e\}, s_1) = \{T_1, T_2, T_4\}$. Thus, the support of pattern $\{a, e\}$ in that sequence is $sup(\{a, e\}, s_1) = |TR(\{a, e\}, s_1)| = 3$ and its periods are $pr(\{a, e\}, s_1) = \{1, 1, 2, 1\}$.

In traditional PFPM [6], a pattern (itemset) is considered as periodic-frequent in a sequence if its maximum period is no greater than a user-specified *maximum period threshold* ($maxPr$), and its support is no less than a *minimum support threshold* ($minSup$). For instance, consider the database of Table 1, $maxPr = 3$ and $minSup = 3$. The itemset $\{a, e\}$ is periodic-frequent in sequence s_1 since its maximum period is $max\{1, 1, 2, 1\} = 2 \leq maxPr$ and $sup(\{a, e\}, s_1) = 3 \geq minSup$. Nonetheless, a major drawback of the $maxPr$ constraint is that a pattern is discarded if it has just one or a few periods greater than $maxPer$. But on the other hand, if a large value is set for $maxPer$, patterns having periods that vary greatly may be output as periodic patterns. To address these issues, several alternative periodicity measures have been considered [7, 19, 8, 9, 10, 11, 12, 13, 14]. However, most of these studies focus on finding periodic patterns with high frequency, and do not consider rare periodic patterns. To address the limitations of the $maxPer$ constraint, this paper proposes to assess a pattern's periodicity using the standard deviation of periods.

Definition 3 (Standard deviation of periods). The average period of a pattern X in a sequence s is $avgPr(X, s) = \frac{\sum_{i=1}^{k+1} per_i}{k+1}$, where $k = sup(X, s)$. The standard deviation of periods of pattern X in s is $stanDev(X, s) = \sqrt{\frac{\sum_{i=1}^{k+1}(per_i - avgPr(X,s))^2}{k+1}}$.

For instance, itemset $\{a, e\}$ has the periods $pr(\{a, e\}, s_1) = \{1, 1, 2, 1\}$ in sequence s_1. Hence, its average period is $avgPr(\{a, e\}, s_1) = (1 + 1 + 2 + 1)/4 = 1.25$, and the standard deviation of its periods is $stanDev(\{a, e\}, s_1) = \sqrt{[(1 - 1.25)^2 + (1 - 1.25)^2 + (2 - 1.25)^2 + (1 - 1.25)^2]/4}$.

Two lemmas allows to efficiently calculate the average and standard deviation of periods.

Lemma 1 (Average period calculation). An alternative way of calculating the average period of an itemset X in a sequence s is $avgPr(X, s) = \frac{|s|}{sup(X,s)+1}$.

Proof 1. The average period of a pattern X in a sequence s is $avgPr(X, s) = \frac{\sum_{i=1}^{k+1} per_i}{k+1}$, where $k = sup(X, s)$. To prove the lemma, we show that $\sum_{i=1}^{k+1} per_i = |s|$ holds:

$$\begin{aligned}\sum_{i=1}^{k+1} per_i &= (g(1) - g(0)) + (g(2) - g(1)) + \cdots + (g(k) - g(k-1)) + (g(k+1) - g(k)) \\ &= -g(0) + (g(1) - g(1)) + (g(2) - g(2)) + \cdots + (g(k) - g(k)) + g(k+1) \\ &= g(k+1) - g(0) = |s|\end{aligned}$$

Lemma 2 (Relationship between standard deviation of periods and support). Let the periods of a pattern X in a sequence s be $pr(X, s) = \{per_1, per_2, \cdots, per_{k+1}\}$, where $k = sup(X, s)$. An alternative way of calculating the standard deviation of periods is $stanDev(X, s) = \sqrt{\frac{\sum_{i=1}^{k+1}(per_i)^2}{k+1} - (\frac{|s|}{k+1})^2}$.

Proof 2. We need to show that $\sum_{i=1}^{k+1}(per_i - avgPr(X, s))^2 = \sum_{i=1}^{k+1}(per_i)^2 - \frac{|s|^2}{(k+1)}$. This is done as follows:

$$\begin{aligned}\sum_{i=1}^{k+1}(per_i - avgPr(X, s))^2 &= \sum_{i=1}^{k+1}(per_i)^2 - 2\sum_{i=1}^{k+1}(per_i * avgPr(X, s)) + \sum_{i=1}^{k+1}(avgPr(X, s))^2 \\ &= \sum_{i=1}^{k+1}(per_i)^2 - (k+1)(avgPr(X, s))^2 && \text{(by Definition 3)} \\ &= \sum_{i=1}^{k+1}(per_i)^2 - \frac{|s|^2}{k+1} && \text{(by Lemma 1)}\end{aligned}$$

Lemmas 1 and 2 are useful to efficiently calculate the standard deviation of periods of itemsets in a sequence. By Lemma 1, if the term $|s|$ is calculated once, the average period of any itemset X in s can then be direclty obtained by dividing $sup(X, s) + 1$ by the result. Using Lemma 2, for each i, time to calculate the difference between per_i and $avgPr(X, s)$ can be saved. Calculating the standard deviation in this way is more efficient than using Definition 3, since this latter requires performing two steps. The above lemmas are also interesting as they show the relationship between average, standard deviation and support for periods in a sequence.

To find rare periodic patterns in a sequence, a simple solution is to discover all patterns having a support no greater than a *maxSup* treshold. However, if this constraint is only used with the standard deviation of periods, many spurious patterns may be output. This is because there is no measure of correlation that items in an itemset are correlated. To address this issue, we integrate the *bond* measure to define the concept of rare correlated periodic patterns [24, 29, 30, 33, 34].

Definition 4 (Disjunctive support). In a sequence s, the disjunctive support of an itemset X is the number of transactions containing at least one item from X, denoted as $dissup(X, s)$.

Definition 5 (Bond). In a sequence s, the *bond* of an itemset X is $bond(X, s) = \frac{sup(X,s)}{dissup(X,s)}$.

For instance, $dissup(\{a, e\}, s_1) = |\{T_1, T_2, T_3, T_4, T_5\}| = 5$ and $bond(\{a, e\}, s_1) = \frac{3}{5} = 0.6$.

Property 1 (Anti-monotonicity of the bond measure). Let there be two itemsets $X \subset Y$. Then, $bond(X) \geq bond(Y)$ [33].

Definition 6 (Rare correlated periodic pattern in a sequence). Let there be three user-specified thresholds *maxSup*, *maxStd*, and *minBond*. An itemset X is a rare correlated periodic pattern in a sequence s if its $sup(X, s) \leq maxSup$, $stanDev(X, s) \leq maxStd$ and $bond(X, s) \geq minBond$.

For instance, if the user sets $maxSup = 2$, $maxStd = 1$ and $minBond = 0.6$, the itemset {b,e} is a rare correlated periodic pattern in sequence s_1, and $sup(\{b, e\}, s_1) = 2$, $stanDev(\{b, e\}, s_1) = 0.47$ and $bond(\{b, e\}, s_1) = 0.67$.

Definition 6 can be used to find patterns in a single sequence. The next definition extends the concept of rare correlated periodic patterns for many sequences based on a novel sequence periodic ratio measure.

Definition 7 (Sequence periodic ratio). In a sequence database SDB, $numSeq(X)$ denotes the number of sequences where an itemset X is a rare correlated periodic pattern. Let $|SDB|$ be the number of sequences in SDB. The ratio $ra(X) = numSeq(X)/|SDB|$ is called the sequence periodic ratio of X in SDB.

For instance, the database of Table 1 contains $|SDB| = 4$ sequences. The number of sequence where {b,e} is a rare correlated periodic pattern is $numSeq(\{b, e\}) = 3$ (s_1, s_2, and s_4). Hence, the sequence periodic ratio of {b,e} is $ra(\{b, e\}) = 3/4 = 0.75$.

Problem Statement. Consider a sequence database SDB and four user-specified thresholds: maximum support threshold (*maxSup*), maximum bond threshold (*maxBond*), maximum standard deviation threshold (*maxStd*) and minimum sequence periodic ratio threshold (*minRa*). An itemset X is a RCPPS in SDB if $ra(X) \geq minRa$. The problem of mining RCPPs common to multiple sequences is to find all RCPPSs.

Discovering RCPPSs in a database is not easy because the maximum support constraint cannot be used for search space pruning, as well as the standard deviation of periods because it is neither monotonic nor anti-monotonic. Hence, to be able to prune the search space, an upper-bound on the *ra* measure is introduced.

Definition 8 (upBondRa). Given a user-specified threshold *minBond*, an itemset X is a candidate in a sequence s if $bond(X, s) \geq minBond$. The number of sequences where an itemset X is a candidate in a sequence database SDB is denoted as $numCand(X)$. The *upBondRa* of X in SDB is defined as $upBondRa(X) = numCand(X)/|SDB|$.

Property 2. For two itemsets $X \subset X'$, (1) $upBondRa(X) \geq ra(X)$ and (2) $upBondRa(X) \geq upBondRa(X')$. Proof is ommitted due to space limitation.

4. The MRCPPS Algorithm

This section presents an algorithm named MRCPPS (Mining Rare Correlated Periodic Patterns common to multiple Sequences) to efficiently enumerate all RCPPS. MRCPPS applies a depth-first search to explore the search space of itemsets. Initially, it considers patterns each having a single item. Then, it recursively appends an item to each itemset to generate larger itemsets. To avoid generating the same itemset more than once, items are appended to itemsets by following a total order $\succ$ on I. Any total order can be used such as the lexicographical order. The search space of itemsets contains up to $2^{|I|}-1$ itemsets (excluding the empty set). To reduce the search space and find RCPPS efficiently, MRCPPS relies on the *upBondRa* measure, which is an upper-bound on the *ra* measure, and satisfies the *downward closure property* (Property 2). Moreover, a key problem to design an efficient algorithm is how to calculate all the measures required to evaluate a pattern without scanning the database numerous times. For this purpose, a novel data structure called **RCPPS-list** is designed. A RCPPS-list is created for each itemset X considered by MRCPPS. The RCPPS-list stores information about an itemset X using three fields: (1) **SIDlist**: is the list of identifiers of sequences containing X. (2) **list-conTIDlist**: contains the list of identifiers of transactions where X occurs (conTIDlist) for each sequence in SIDlist. (3) **list-disTIDlist**: is the list of identifiers of transactions containing at least one item from X (disTIDlist) for each sequence in SIDlist. For instance, the RCPPS-list of the itemset $\{b, e\}$ is: *SIDlist* : [1, 2, 4], *list-conTIDlist* : {[2, 4], [2, 4], [1, 4]}, *list-disTIDlist* : { [1, 2, 4], [2, 4, 5], [1, 2, 4]}. Using the information stored in an RCPPS-list, MRCPPS can quickly compute the *support* of a pattern in a sequence (it is the size of the *conTIDlist* corresponding to the sequence ID), its *bond* (it is the size of its *conTIDlist* divided by the size of its *disTIDlist*, for that sequence ID), and *stanDev* (it can be calculated from the *conTIDlist* corresponding to the sequence ID). For example, the RCPPS-list of the itemset $\{b, e\}$ indicates that this latter appears in sequence 1, 2, and 4, and thus that it has a support of 3. The *conTIDlist* and *disTIDlist* of itemset $\{b, e\}$ for sequence 1 are [2, 4] and [1, 2, 4], respectively. Thus, the bond of $\{b, e\}$ in sequence 1 is 2 / 3. After calculating the *support*, *bond* and *stanDev* measures of an itemset, its *upBondRa* and *ra* values can be quickly calculated to check if the itemset should be output as a RCPPS and if it is should be used to generate larger patterns (according to Property 2). Moreover, as it will be explained, the RCPPS-list of any itemset containing more than one item can be obtained without scanning the database by performing intersections of the *SIDlists* and *conTIDlists* of two of its subsets and the union of their *disTIDlists*.

The proposed MRCPPS (Algorithm 1) takes as input a sequence database and the user-defined $maxSup$, $maxStd$, $minBond$ and $minRa$ thresholds. MRCPPS outputs all the RCPPS. It first scans the database to create the RCPPS-list of each item. Then, MRCPPS identifies the set I^* of all items having a *upBondRa* value that is no less than $minRa$ (other items are ignored since they cannot be part of a RCPPS by Property 2). The *upBondRa* values of items are then used to establish a total order $\succ$ on items, which is the ascending order of *upBondRa* values. Finally, the depth-first search exploration of itemsets starts by calling the recursive *Search* procedure with the empty itemset $\emptyset$, the set of single items I^*, $maxSup$, $maxStd$, $minBond$ and $minRa$.

The *Search* procedure is presented in Algorithm 2. It has the following parameters: (1) an itemset P, (2) extensions of the form Pz obtained by adding an item z to P, and where $upBondRa(Pz) \geq minRa$, (3) the sequence database SDB (4) $maxSup$, (5) $maxStd$, (6) $minBond$ and (7) $minRa$. When the procedure is initially called, $P = \emptyset$ and extensions of P are single items ($ExtensionsOfP = I$). The procedure iterates over each extension Px of P (line 1 to 15). For an extension, the procedure calculates $numSeq(Px)$ and $ra(Px)$ using the RCPPS-list of Px (line 2 to 3). If $ra(Px) \geq minRa$, then Px is a RCPPS and it is output (line 4). Note that if the *upBondRa* value of Px is no less than $minRa$, the extensions of Px should be explored (no matter if Px is a RCPPS or not). This is performed by merging Px with all extensions Py of P such that y $\succ x$ to form extensions of the form Pxy containing $|Px| + 1$ items (line 7). The RCPPS-list of Pxy is then constructed by calling the *Construct* procedure using the RCPPS-list of Px and Py (line 8). An extension Pxy is then added to the set of extensions to be considered by the depth-first search if the *upBondRa* value of Pxy is no less than $minRa$ (line 11 to 13). This has for effect of ignoring some extensions from the search space that cannot be RCPPS. This search space reduction strategy can be done because the *upBondRa* measure is anti-monotonic (Property 2). Hence, any extensions of a non RCPPS is

Algorithm 1: The MRCPPS algorithm

input : SDB: a sequence database,
$maxSup$, $maxStd$, $minBond$, $minRa$: the user-specified threshods.
output: the set of RCPPS

1 Scan SDB to calculate the RCPPS-lists of all items in I;
2 $I^* \leftarrow \emptyset$;
3 **foreach** *item* $i \in I$ **do**
4 $\quad numCand(i) \leftarrow |\{s|bond(i,s) \geq minBond \wedge s \in SDB\}|$;
5 $\quad upBondRa(i) \leftarrow numCand(i)/|SDB|$;
6 $\quad$ **if** $upBondRa(i) \neq 0 \wedge upBondRa(i) \geq minRa$ **then** $I^* \leftarrow I^* \cup \{i\}$;
7 **end**
8 Sort I^* by the order $\succ$ of ascending $upBondRa$ values;
9 `Search` ($\emptyset$, I^*, SDB, $minSup$, $maxStd$, $minBond$, $minRa$);

safely eliminated from the search space. Thereafter, the *Search* procedure is called with Pxy to calculate its ra and explore its extensions(s) using the depth-first search (line 16).

Algorithm 2: The Search procedure

input : P: an itemset, $ExtensionsOfP$: a set of extensions of P, SDB: the sequence database
$maxSup$, $maxStd$, $minBond$, $minRa$: the user-specified threshods.
output: the set of RCPPS

1 **foreach** *itemset* $Px \in ExtensionsOfP$ **do**
2 $\quad numSeq(Px) \leftarrow |\{s|sup(Px,s) \leq maxSup \wedge stanDev(Px,s) \leq maxStd \wedge bond(Px,s) \geq minBond \wedge s \in SDB\}|$;
3 $\quad ra(Px) \leftarrow numSeq(Px)/|SDB|$;
4 $\quad$ **if** $ra(Px) \neq 0 \wedge ra(Px) \geq minRa$ **then** output Px;
5 $\quad ExtensionsOfPx \leftarrow \emptyset$;
6 $\quad$ **foreach** *itemset* $Py \in ExtensionsOfP$ *such that* $y \succ x$ **do**
7 $\quad\quad Pxy \leftarrow Px \cup Py$;
8 $\quad\quad Pxy.RCPPS\text{-}list \leftarrow$ `Construct`$(Px.RCPPS\text{-}list, Py.RCPPS\text{-}list)$;
9 $\quad\quad numCand(Pxy) \leftarrow |\{s|bond(Pxy,s) \geq minBond \wedge s \in SDB\}|$;
10 $\quad\quad upBondRa(Pxy) \leftarrow numCand(Pxy)/|SDB|$;
11 $\quad\quad$ **if** $upBondRa(Pxy) \neq 0 \wedge upBondRa(Pxy) \geq minRa$ **then**
12 $\quad\quad\quad ExtensionsOfPx \leftarrow ExtensionsOfPx \cup Pxy$;
13 $\quad\quad$ **end**
14 $\quad$ **end**
15 **end**
16 `Search` (Px, $ExtensionsOfPx$, SDB, $maxSup$, $maxStd$, $minBond$, $minRa$);

The *Construct* procedure takes as parameters the RCPPS-list $listPx$ and $listPy$ of some itemsets Px and Py, and it returns the RCPPS-list of the itemset Pxy. The procedure initializes an empty RCPPS-list $listPxy$ for Pxy (line 1). Then, a loop iterates over each sequence that appears in both $SIDlist$ of Px and Py. For each such sequence, let i, j be the index of $SIDlist$ of Px and Py that respectively represent the same sequence. Those indexes are used to retrieve the *list-conTIDlist* and *list-disTIDlist* of Px and Py, respectively. Then, the *list-conTIDlist* of Pxy can be obtained by intersecting *list-conTIDlist*(i) and *list-conTIDlist*(j). And the *list-disTIDlist* of Pxy can be obtained by joining *list-disTIDlist*(i) and *list-disTIDlist*(j). Then procedure returns the RCPPS-list of Pxy, which is obtained without scanning the database.

Algorithm 3: The Construct procedure

input : $listPx$: the RCPPS-list of Px, $listPy$: the RCPPS-list of Py.
output: the RCPPS-list of Pxy

$listPxy \leftarrow \emptyset$;
foreach *i, j where listPx.SIDlist(i) = listPy.SIDlist(j)* **do**
  $conTIDlist \leftarrow listPx.list\text{-}conTIDlist(i) \cap listPy.list\text{-}conTIDlist(j)$;
  if $conTIDlist \neq \emptyset$ **then**
    $disTIDlist \leftarrow listPx.list\text{-}disTIDlist(i) \cup listPy.list\text{-}disTIDlist(j)$;
    $listPxy.SIDlist \leftarrow listPxy.SIDlist \cup listPx.SIDlist(i)$;
    $listPxy.list\text{-}conTIDlist \leftarrow listPxy.list\text{-}conTIDlist \cup conTIDlist$;
    $listPxy.list\text{-}disTIDlist \leftarrow listPxy.list\text{-}disTIDlist \cup disTIDlist$;
  end
end
return $listPxy$;

4.1. A Detailed Example

A detailed example is provided to illustrate how the algorithm is applied. Consider that the MRCPPS algorithm is applied with $minSup = 2$, $maxStd = 1$, $minBond = 0.6$ and $minRa = 0.6$, on the example database of Table 1. The main procedure (Algorithm 1) is called, and the algorithm starts by processing single items. Consider the item $\{b\}$. MRCPPS reads the database and finds that the RCPPS-list of $\{b\}$ contains: (1) *SIDlist*: [1, 2, 3, 4], (2) *list-conTIDlist*: {[2, 4], [2, 4, 5], [1, 2, 5], [1, 2, 4]} and (3) *list-disTIDlist*: {[2, 4], [2, 4, 5], [1, 2, 5], [1, 2, 4]}. Moreover, the *upBondRa* value of $\{b\}$ is 1.0. Thus, extensions of $\{b\}$ will be considered. Other items are treated in a similar way and it is found that extensions of items $\{a\}$, $\{c\}$, $\{d\}$ and $\{e\}$ should also be considered. The RCPPS-list of these items are built, and the *Search* procedure (Algorithm 2) is called to find RCPPS. Since the *ra* of $\{b\}$ is 0.25, $\{b\}$ is not a RCPPS. Then, extensions of $\{b\}$ need to be explored. Consider the extension of $\{b\}$ with $\{e\}$, that is $\{b, e\}$. The *Construct* procedure (Algorithm 3) is applied on the RCPPS-list of $\{b\}$ and $\{e\}$ to generate the RCPPS-list of $\{b, e\}$. For this, the sequences where $\{b\}$ and $\{e\}$ are both periodic and have a *upBondRa* value no less than *minRa* are found. The RCPPS-list of $\{e\}$ is: (1) *SIDlist*: [1, 2, 4], (2) *list-conTIDlist*: {[1, 2, 4], [2, 4], [1, 4]}, (3) *list-disTIDlist*: {[1, 2, 4], [2, 4], [1, 4]}. Thus, the RCPPS-list of $\{b, e\}$ is: (1) *SIDlist*: [1, 2, 4], (2) *list-conTIDlist*: {[2, 4], [2, 4], [1, 4]}, (3) *list-disTIDlist*: {[1, 2, 4], [2, 4, 5], [1, 2, 4]}. The RCPPS-list of $\{b, e\}$ is returned by the *Search* procedure. The *upBondRa* value of $\{b, e\}$ is 0.75. Thus, the *Search* procedure is next called to explore extensions of $\{b, e\}$. It is found that the *ra* of $\{b, e\}$ is 0.75. Thus $\{b, e\}$ is a RCPPS and it is output. The same process is repeated for other itemsets by recursively calling the *Search* procedure until all RCPPSs are found. Because the algorithm can explore the search space of all itemsets, and it only eliminates itemsets using Property 2, no RCPPS can be missed, and the algorithm is complete.

5. Experimental Evaluation

Two algorithms named PHUSPM [14] and MPFPS [15] were proposed to mine periodic frequent patterns in a sequence database. However, they are not designed for mining rare correlated patterns. Hence, the performance of these algorithms cannot be compared with the proposed MRCPPS algorithm. To evaluate the performance of MRCPPS, this section compares its behavior for several parameter values, including with a baseline version of MRCPPS that finds all rare patterns. MRCPPS is implemented in Java, and it was run on a Windows 10 computer equipped with a 3.60 GHz Xeon E3 processor with 64 GB of RAM. Three benchmark datasets (FIFA, Bible and Leviathan) were used in the experiments, obtained from the SPMF data mining library [35]. The first one is click-stream data from a website, while the two latter are sequences of words from books. Since the number of items per transaction in these databases is always one,

it was decided to group items from consecutive transactions in a same transaction (by three items for FIFA and Bible, and ten for Leviathan). This allows to evaluate the algorithm's performance on sequences where transactions contain multiple items. Let $|SDB|$, $|I|$, $|T|$, S_{min}, S_{max}, S_{avg} be the number of sequences, distinct items, item count per transaction, min, max and average transaction count per sequence. The characteristics of the datasets are presented in Table 2. The source code of MRCPPS can be obtained from `http://www.philippe-fournier-viger.com/spmf/`.

Table 2: Characteristics of the datasets

Dataset	$\|SDB\|$	$\|I\|$	$\|T\|$	S_{min}	S_{max}	S_{avg}
FIFA	20,450	2,990	3	3	34	12.4
Bible	36,369	13,905	3	3	34	7.5
Leviathan	5,834	9,025	10	1	10	3.8

MRCPPS provides four parameters: *maxSup*, *maxStd*, *minBond* and *minRa*. The first one is used to define what is a rare pattern in terms of frequency. This parameter is set to a low value in the experiments to find rare patterns ($maxSup = 5$). The second and fourth parameters are used to select periodic patterns, while the third one is used to identify correlated patterns. In the special case where *maxStd* is set to ∞ and *minBond* and *minRa* are set to 0, the algorithm finds all rare patterns. In the following, this setting is used as baseline, and is denoted as $baseline(0, 0, 100)$. The baseline is compared with other parameter settings to assess the influence of the *minRa*, *minBond* and *maxStd* parameters on the performance of the algorithm and number of patterns found. In the following $MRCPPS(a, b, c)$ denotes MRCPPS with $minRa = a$, $minBond = b$, $maxStd = c$ and $maxSup = 5$.

5.1. Influence of minBond

The *minBond* parameter was first varied to assess its impact on performance. Figure 1 depicts the runtime and number of patterns found for different *minBond* values. For each value x of *minBond*, three versions of MRCPSS are compared with the baseline. $MRCPPS(0.1\%, x, 100)$ means to mine rare patterns common to multiple sequences, $MRCPPS(0, x, 1)$ means to mine rare correlated patterns, while $MRCPPS(0.1\%, x, 1)$ refers to mining RCPPSs.

In general, it is observed that mining RCPPS is much faster than mining all rare patterns using the baseline algorithm. This is what we expected since the algorithm uses the *upBondRa* upper-bound for pruning and reducing the search space. Since *upBondRa* is calculated using the *bond* value of each sequence, the runtime and number of patterns decrease or stay the same as *minBond* is increased. This can be clearly observed in most charts of Figure 1. However, for the FIFA and Bible datasets, the runtime does not change much as *minBond* is increased. This is reasonable because the more items a pattern contains, the smaller the *bond* usually is (see Definition 5). Thus, the bond measure is more effective to prune large patterns than smaller patterns. Hence, since the maximum item count per transaction in these two datasets is three, the *minBond* threshold does not have a strong influence on the number of patterns found. The bottommost charts of Figure 1 show that as *minBond* is increased, the runtime and the number of patterns greatly decrease. It is reasonable that the *minBond* threshold has this effect on the Leviathan dataset since itemsets contain up to ten items. Hence, there are many opportunities to generate large patterns with low bond in the Leviathan dataset. Note that in Fig. 1, the symbol k denotes thousands and the symbol m denotes millions.

5.2. Influence of minRa

Next, the *minRa* threshold was varied to evaluate how it influences performance. In Figure 2, the pattern count, and peak memory usage of the algorithm is reported for different *minRa* values. Note that for the three subfigures on the left, the charts are enlarged to better show differences for high *minRa* values. Moroever, the symbol x in Figure 2 represents the variable that is varied (*minRa*).

Two observations are drawn from these results: (i) Decreasing *minRa* tends to increase the number of patterns found and memory usage. The reason is that as *minRa* is decreased, it becomes more difficult

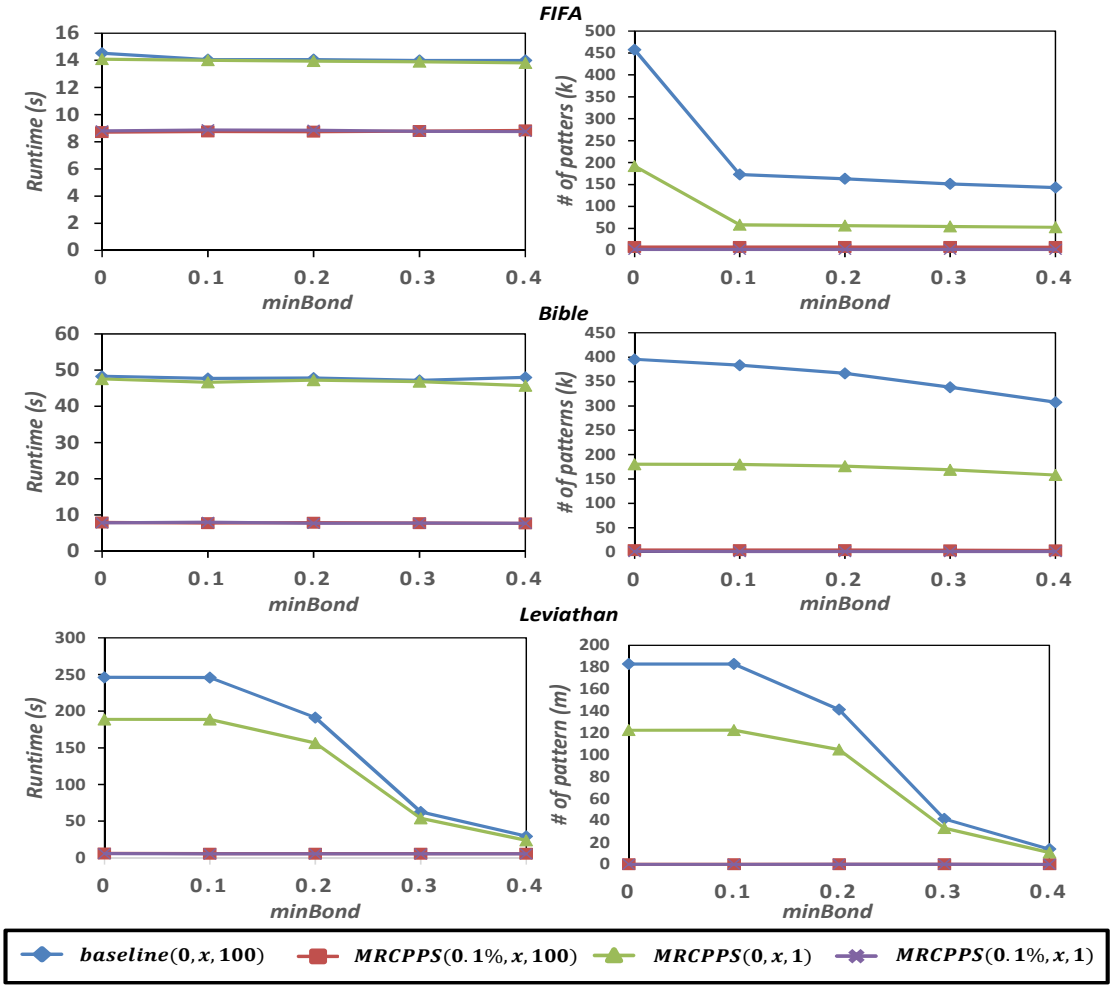

Figure 1: Runtimes and number of patterns for different *minBond* values.

to reduce the search space using *upBondRa*, since more patterns may have *ra* values that are no less than *upBondRa*. Thus, the algorithm may consider more patterns in the search space, which increases the runtime. Although detailed results for the runtime are not shown for this experiment due to space limitation, this behavior can be observed in Figure 1. (*ii*) The *minRa* threshold has a greater effect on memory consumption for the datasets that have many items and few transactions (Bible and Leviathan). When *minRa* is increased from 0 to 0.05%, memory consumption is reduced to one eighth of the initial value. This is reasonable because the item count per transaction is fixed, and many items do not appear in sequences having few transactions. Hence, using the *minRa* constraint, memory can be saved by avoiding considering itemsets that do not appear in multiple sequences.

5.3. Influence of maxStd

Then, the *maxStd* parameter was varied to evaluate its influence. Table 3 shows the number of patterns found for different *maxStd* values on the FIFA and Bible datasets (results are not shown for Leviathan but a similar trend is observed). It can be observed that increasing *maxStd* often results in finding more patterns. The reason is that increasing *maxStd* increases the range of *stanDev* values that are allowed for a pattern to be considered as periodic in a sequence. It is further observed that as *maxStd* is decreased from 2 to 0 in the Table, the number of patterns decreases in some cases to about one tenth of the initial

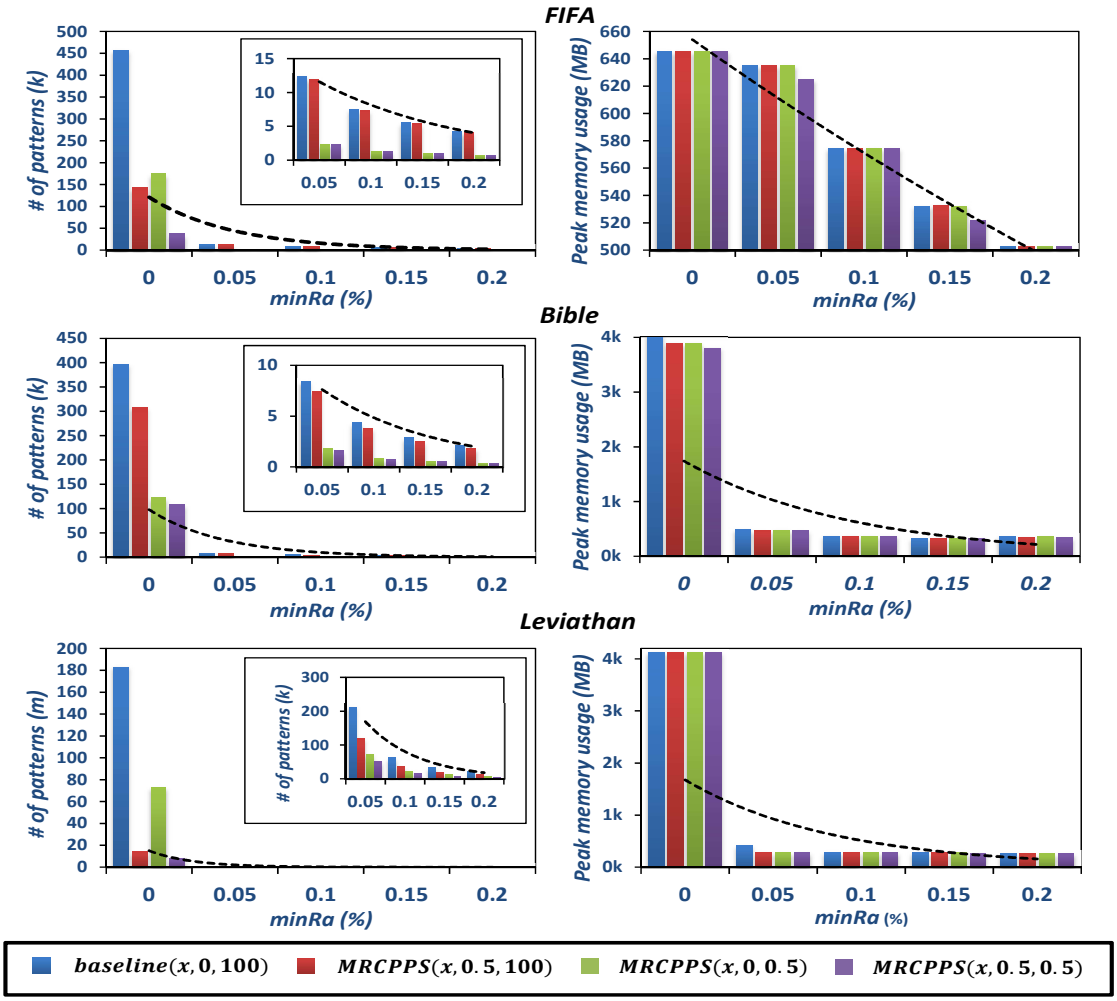

Figure 2: Number of patterns and peak memory usage for different *minRa* values.

number. Moreover, if the *minBond* and *minRa* constraints are both used at the same time, the number of patterns can be one thousand times smaller than the initial number. It shows that these parameters can be used together to achieve better performance and filter many non correlated and non periodic patterns.

6. Conclusion

This article defined a novel problem of mining rare correlated periodic patterns that appear in multiple sequences. The constraints of maximum support, minimum bond, maximum standard deviation and minimum sequence periodic ratio have been used, and properties of these measures have been studied. To efficiently enumerate all RCPPS using these measures, an efficient algorithm named RMCPPS was proposed, based on a novel RCPPS-list structure and a novel upBondRa upper-bound to reduce the search space. Experiments on several real databases have shown that the designed algorithm is efficient and can filter many non periodic or non correlated patterns.

The research presented in this paper sets forward several possibilities for future work. First, one could design more efficient algorithms in terms of runtime and memory consumption. Second, one could consider adapting the proposed model for discovering other types of patterns such as periodic sequential rules in sequences. Third, methods could be developed to automatically tune parameters for a given dataset.

Table 3: Number of patterns found by MRCPPS for different $maxStd$ values on FIFA and Bible.

datasets	# of patterns / Algorithm \ $maxStd$	0	0.5	1	1.5	2
FIFA	$baseline(0, 0, x)$	21,639	174,770	192,049	207,915	220,463
	$MRCPPS(0, 0.5, x)$	17,935	37,328	52,628	64,845	74,498
	$MRCPPS(0.1\%, 0, x)$	490	1,347	2,081	2,670	3,164
	$MRCPPS(0.1\%, 0.5, x)$	488	1,321	2,050	2,622	3,100
Bible	$baseline(0, 0, x)$	52,579	122,290	180,379	227,503	263,774
	$MRCPPS(0, 0.5, x)$	47,126	108,166	158,224	197,791	226,833
	$MRCPPS(0.1\%, 0, x)$	252	832	1,503	2,088	2,586
	$MRCPPS(0.1\%, 0.5, x)$	233	766	1,377	1,911	2,353

References

[1] J. Han, J. Pei, Y. Yin, R. Mao, Mining frequent patterns without candidate generation: A frequent-pattern tree approach, Data Min. Knowl. Discov. 8 (1) (2004) 53–87. doi:10.1023/B:DAMI.0000005258.31418.83.

[2] R. Agrawal, R. Srikant, Fast algorithms for mining association rules in large databases, in: Proceedings of 20th International Conference on Very Large Data Bases, 1994, pp. 487–499.

[3] M. J. Zaki, Scalable algorithms for association mining, IEEE Trans. Knowl. Data Eng. 12 (3) (2000) 372–390. doi:10.1109/69.846291.

[4] C. C. Aggarwal, J. Han (Eds.), Frequent Pattern Mining, Springer, 2014. doi:10.1007/978-3-319-07821-2.

[5] P. Fournier-Viger, J. C. Lin, B. Vo, T. C. Truong, J. Zhang, H. B. Le, A survey of itemset mining, Wiley Interdiscip. Rev. Data Min. Knowl. Discov. 7 (4) (2017). doi:10.1002/widm.1207.

[6] S. K. Tanbeer, C. F. Ahmed, B. Jeong, Y. Lee, Discovering periodic-frequent patterns in transactional databases, in: Proc. 13th Pacific-Asia Conference on Knowledge Discovery and Data Mining, 2009, pp. 242–253. doi:10.1007/978-3-642-01307-2_24.

[7] K. Amphawan, P. Lenca, A. Surarerks, Mining top-K periodic-frequent pattern from transactional databases without support threshold, in: Proc. Third International Conference on Advanced in Information Technology, 2009, pp. 18–29. doi:10.1007/978-3-642-10392-6_3.

[8] A. Surana, R. U. Kiran, P. K. Reddy, An efficient approach to mine periodic-frequent patterns in transactional databases, in: Proc. 15th Pacific-Asia Conference on Knowledge Discovery and Data Mining, 2011, pp. 254–266. doi:10.1007/978-3-642-28320-8_22.

[9] M. M. Rashid, M. R. Karim, B. Jeong, H. Choi, Efficient mining regularly frequent patterns in transactional databases, in: Proc. 17th International Conference on Database Systems for Advanced Applications Part I, 2012, pp. 258–271. doi:10.1007/978-3-642-29038-1_20.

[10] K. Amphawan, P. Lenca, A. Surarerks, Mining top-K periodic-frequent pattern from transactional databases without support threshold, in: Proc. of Third International Conference on Advances in Information Technology, 2009, pp. 18–29. doi:10.1007/978-3-642-10392-6_3.

[11] R. U. Kiran, M. Kitsuregawa, P. K. Reddy, Efficient discovery of periodic-frequent patterns in very large databases, Journal of Systems and Software 112 (2016) 110–121. doi:10.1016/j.jss.2015.10.035.

[12] P. Fournier-Viger, J. C. Lin, Q. Duong, T. Dam, PHM: mining periodic high-utility itemsets, in: Proc. 16th Industrial Conference on Data Mining, 2016, pp. 64–79. doi:10.1007/978-3-319-41561-1_6.

[13] J. N. Venkatesh, R. U. Kiran, P. K. Reddy, M. Kitsuregawa, Discovering periodic-frequent patterns in transactional databases using all-confidence and periodic-all-confidence, in: Proc. 27th International Conference on Database and Expert Systems Applications Part I, 2016, pp. 55–70. doi:10.1007/978-3-319-44403-1_4.

[14] D. Dinh, B. Le, P. Fournier-Viger, V. Huynh, An efficient algorithm for mining periodic high-utility sequential patterns, Appl. Intell. 48 (12) (2018) 4694–4714. doi:10.1007/s10489-018-1227-x.

[15] P. Fournier-Viger, Z. Li, J. C. Lin, R. U. Kiran, H. Fujita, Discovering periodic patterns common to multiple sequences, in: Proc. 20th International Conference on Big Data Analytics and Knowledge Discovery, 2018, pp. 231–246. doi:10.1007/978-3-319-98539-8_18.

[16] A. C. M. Fong, B. Zhou, S. C. Hui, G. Y. Hong, T. Do, Web content recommender system based on consumer behavior modeling, IEEE Trans. Consumer Electronics 57 (2) (2011) 962–969. doi:10.1109/TCE.2011.5955246.

[17] E. F. Glynn, J. Chen, A. R. Mushegian, Detecting periodic patterns in unevenly spaced gene expression time series using lomb-scargle periodograms, Bioinformatics 22 (3) (2006) 310–316. doi:10.1093/bioinformatics/bti789.

[18] F. Yi, L. Yin, H. Wen, H. Zhu, L. Sun, G. Li, Mining human periodic behaviors using mobility intention and relative entropy, in: Proc. 22nd Pacific-Asia Conference on Knowledge Discovery and Data Mining, 2018, pp. 488–499. doi:10.1007/978-3-319-93034-3_39.

[19] R. U. Kiran, P. K. Reddy, Mining rare periodic-frequent patterns using multiple minimum supports, in: Proceedings of the 15th International Conference on Management of Data, 2009.

[20] P. Fournier-Viger, J. C.-W. Lin, T. Truong-Chi, R. Nkambou, A survey of high utility itemset mining, in: High-Utility Pattern Mining, Springer, 2019, pp. 1–45.

[21] P. Fournier-Viger, Y. Zhang, J. C.-W. Lin, H. Fujita, Y. S. Koh, Mining local and peak high utility itemsets, Information Sciences 481 (2019) 344–367.

[22] P. Fournier-Viger, P. Y. J. C. Lin, U. Kiran, Discovering stable periodic-frequent patterns in transactional data, in: Proc. 32nd Intern. Conf. on Industrial, Engineering and Other Applications of Applied Intelligent Systems, 2019.

[23] E. Cohen, M. Datar, S. Fujiwara, A. Gionis, P. Indyk, R. Motwani, J. D. Ullman, C. Yang, Finding interesting associations without support pruning, IEEE Trans. Knowl. Data Eng. 13 (1) (2001) 64–78. doi:10.1109/69.908981.

[24] E. Omiecinski, Alternative interest measures for mining associations in databases, IEEE Trans. Knowl. Data Eng. 15 (1) (2003) 57–69. doi:10.1109/TKDE.2003.1161582.

[25] D. T. J. Huang, Y. S. Koh, G. Dobbie, Rare pattern mining on data streams, in: Proc. 14th International Conference on Data Warehousing and Knowledge Discovery, 2012, pp. 303–314. doi:10.1007/978-3-642-32584-7_25.

[26] Y. S. Koh, S. D. Ravana, Unsupervised rare pattern mining: A survey, ACM Transactions on Knowledge Discovery 10 (4) (2016) 45:1–45:29. doi:10.1145/2898359.

[27] S. Bouasker, T. Hamrouni, S. B. Yahia, New exact concise representation of rare correlated patterns: Application to intrusion detection, in: Proc. 16th Pacific-Asia Conference on Knowledge Discovery and Data Mining, pages = 61–72, year = 2012, crossref = DBLP:conf/pakdd/2012-2, doi = 10.1007/978-3-642-30220-6_6, timestamp = Thu, 25 May 2017 00:42:26 +0200, biburl = https://dblp.org/rec/bib/conf/pakdd/BouaskerHY12, bibsource = dblp computer science bibliography, https://dblp.org.

[28] S. Bouasker, S. B. Yahia, Inferring knowledge from concise representations of both frequent and rare jaccard itemsets, in: Proc. 24th International Conference on Database and Expert Systems Applications, 2013, pp. 109–123. doi:10.1007/978-3-642-40173-2_12.

[29] S. Bouasker, S. B. Yahia, Key correlation mining by simultaneous monotone and anti-monotone constraints checking, in: Proceedings of the 30th Annual ACM Symposium on Applied Computing, pages = 851–856, year = 2015, crossref = DBLP:conf/sac/2015, doi = 10.1145/2695664.2695802, timestamp = Tue, 06 Nov 2018 11:06:47 +0100, biburl = https://dblp.org/rec/bib/conf/sac/BouaskerY15, bibsource = dblp computer science bibliography, https://dblp.org.

[30] A. Soulet, C. Raïssi, M. Plantevit, B. Crémilleux, Mining dominant patterns in the sky, in: Proc. 11th IEEE International Conference on Data Mining, 2011, pp. 655–664. doi:10.1109/ICDM.2011.100.

[31] C. F. Ahmed, S. K. Tanbeer, B. Jeong, H. Choi, A framework for mining interesting high utility patterns with a strong frequency affinity, Inf. Sci. 181 (21) (2011) 4878–4894. doi:10.1016/j.ins.2011.05.012.

[32] M. Barsky, S. Kim, T. Weninger, J. Han, Mining flipping correlations from large datasets with taxonomies, CoRR abs/1201.0233 (2012). arXiv:1201.0233.

[33] N. B. Younes, T. Hamrouni, S. B. Yahia, Bridging conjunctive and disjunctive search spaces for mining a new concise and exact representation of correlated patterns, in: Proc. 13th International Conference on Discovery Science, 2010, pp. 189–204. doi:10.1007/978-3-642-16184-1_14.

[34] P. Fournier-Viger, J. C. Lin, T. Dinh, H. B. Le, Mining correlated high-utility itemsets using the bond measure, in: Proc. 11th Intern. Conf. on Hybrid Artificial Intelligent Systems, 2016, pp. 53–65. doi:10.1007/978-3-319-32034-2_5.

[35] P. Fournier-Viger, J. C.-W. Lin, A. Gomariz, T. Gueniche, A. Soltani, Z. Deng, H. T. Lam, The spmf open-source data mining library version 2, in: Proc. 19th European Conference on Principles of Data Mining and Knowledge Discovery Part III, Springer, 2016, pp. 36–40.