# Finding Periodic Patterns in Multiple Sequences

Philippe Fournier-Viger, Tin Truong Chi, Youxi Wu, Jun-Feng Qu,
Jerry Chun-Wei Lin, Zhitian Li

**Abstract** Discovering periodic patterns in data is an important data analysis task. A periodic pattern is a set of values that regularly appear together over time. Finding such patterns can be useful to understand the data and make predictions. However, most studies on periodic pattern mining have focused on identifying periodic patterns in a single discrete sequence. But for many applications, it is desirable to find patterns that are common to multiple sequences. For instance, it can be interesting to identify periodic behavior that is common to several customers in a store. This chapter gives an overview of the general problem of discovering periodic patterns in multiple sequences, and two special cases that are to find (1) frequent patterns and (2) rare correlated patterns in multiple sequences. Two algorithms are described, which can be applied in many domains where data can be modelled as discrete sequences such as to find periodic patterns in sequences of customer purchases, in sequences of words in a text document, and in sequences

Philippe Fournier-Viger
e-mail: philfv8@yahoo. com

Tin Truong Chi
University of Dalat, Dalat, Vietnam, e-mail: tintc@dlu.edu.vn

Youxi Wu
Hebei University of Technology, Tianjin, China, e-mail: wuc567@163.com

Jun-Feng Qu
School of Computer Engineering, Hubei University of Arts and Science, Xiangyang, China

Jerry Chun-Wei Lin
Department of Computing, Mathematics, and Physics, Western Norway University of Applied Sciences (HVL), Bergen, Norway e-mail: jerrylin@ieee.org

Zhitian Li
Harbin Institute of Technology (Shenzhen), Shenzhen, China, e-mail: tymonlee1@163.com

of treatments received by hospital patients. Research opportunities are also discussed.

# 1 Introduction

*Pattern mining* is an important subfield of data mining that aims at finding interesting patterns in databases [1, 46]. A pattern can be described generally as some association between values appearing in data that can help understanding the data and/or support decision-making. Algorithms have been designed to find patterns in different types of data such as customer transaction databases [21, 25], graphs [36], trees [9, 69], process logs [8], trajectories [15, 70], sequences [24], spatial data [55] and ranking data [35]. Moreover algorithms were proposed to find various types of patterns in data such as frequent patterns [2, 21, 24], rare patterns [39] and periodic patterns [4, 29, 60]

In the last decade, *periodic pattern mining* [4, 5, 29, 37, 38, 57, 60, 62] has emerged as an important data analysis task. It aims at finding patterns that regularly appear in the data. The traditional input of periodic pattern mining is a customer transaction database where each record contains a set of items purchased together at a given time by a customer [4, 29]. In that model, a periodic pattern can be for example that the customer regularly buys wine with cheese (e.g. every week-end). Finding such patterns can be useful for purposes such as marketing or product recommendation. For example, an online store could provide discounts on wine with cheese to a customer that regularly buys these items together.

Generally, a customer transaction database can be viewed as a sequence of discrete events or symbols that are sequentially ordered. Discrete sequences can be found in many domains besides shopping. For example, the sequence of words in a book can be viewed as a discrete sequence, as well as the sequence of locations visited by a person when driving a car. Hence, periodic pattern mining can be applied in many domains.

Several variations of the problem of periodic pattern mining have been studied but the majority of them focuses on analyzing a single discrete sequence. However, for many applications it is desirable to analyze more than a single sequence at a time [19, 28]. For example, continuing the example of analyzing customer transactions, a business owner may want to analyze sequences of transactions made by several customers to discover periodic behavior that is common to several customers rather than analyzing each customer separately. This can provide insights such that many customers regularly buy bread and milk together.

Recently, a few algorithms have been designed to find periodic patterns common to multiple sequences such as MRCPPS [28], MPFPS [19]. These algorithms utilize different definitions of periodic patterns and also of sequences. This chapter gives an overview of the task of mining periodic patterns in a set of discrete sequences, the algorithms, and discuss applications and research opportunities.

The chapter is organized as follows. Section 2 briefly reviews the traditional problem of periodic pattern mining in sequence. Then, section 3 presents the main model for discovering periodic patterns in multiple discrete sequences. Then, Section 4 briefly describes strategies used by the algorithms for this problem. Then, Section 5 discusses opportunities for research on this topic. Finally, Section 6 draws a conclusion.

## 2 Finding Periodic Patterns in a Single Discrete sequence

The type of data considered in traditional periodic pattern mining studies is a discrete sequence [27], also called a sequence of transactions or transaction database, or a transaction database [29].

**Definition 1 (Discrete sequence)** Let there be a set of items $I$ (symbols or event types). An itemset $X$ is a subset of $I$, that is $X \subseteq I$. An itemset containing $k$ items is said to be a $k$-itemset. A sequence $s$ is an ordered list of itemsets $s = \langle T_1, T_2, \ldots T_m \rangle$, where $T_j \subseteq I$ $(1 \leq j \leq m)$, $j$ is the transaction identifier of $T_j$, and $T_j$ is said to be a transaction.

*Example 1* Let there be some products called $a$, $b$, $c$, $d$ and $e$, that are sold in a retail store. This set of products is denoted as $I = \{a, b, c, d, e\}$. Then, a sequence of purchases made by a customer over time can be represented as in Fig. 1. This sequence indicates that a person has bought items $a$, $b$ and $c$ at the same time, then purchased $b$ and $d$ together, then bought $a$, $b$ and $e$ together, then purchased $c$, then bought $a$, $b$, $d$ and $e$ together and so on. This sequence contains eight itemsets. The itemset $\{a, b, c\}$ is a 3-itemsets.

Sequences of transactions can be used to represent data from multiple domains besides shopping data. For example, a sentence in a text can be viewed as a discrete sequence where each word is an item. Another example is a sequence of locations visited by a tourist in a city, where items represent locations. Another example of a discrete sequence is the genome of a virus such as $\langle \{A\}, \{T\}, \{A\}, \{A\}, \{C\}, \{A\}, \ldots \rangle$, where each item is a nucleotide denoted as $A$, $C$, $G$ and $T$ [48]. Thus, as it can be observed by these examples, items in a discrete sequence can be ordered by time or other criteria. A discrete sequence where each transaction contains a single item is called a *string*, and a discrete sequence ordered by time can be called a *temporal database*, *temporal sequence*, *event sequence* [6, 31] or *event log* [32], depending on the context.

To find interesting patterns in a discrete sequence, it is necessary to define more clearly what we mean by periodic pattern and also to define some

$$\boxed{\langle\{a,b,c\},\{b,d\},\{a,b,e\},\{c\},\ \{a,b,d,e\},\{a,c\},\{b,c\},\{b,e\}\rangle}$$

Fig. 1: A discrete sequence of customer purchases

measures to select patterns that are periodic. In the traditional periodic pattern mining task, the type of patterns to be discovered is an itemset. The measures for identifying periodic itemsets are defined based on the following definitions.

**Definition 2 (Subsequence)** Let there be two sequences $s_a = \langle A_1, A_2, \ldots, A_k\rangle$ and $s_b = \langle B_1, B_2, ..., B_l\rangle$. The sequence $s_a$ is a subsequence of $s_b$ if and only if some integer numbers $1 \leq i1 < i2 < ... < ik \leq m$ can be found such that $A_1 \subseteq B_{i1}, A_2 \subseteq B_{i2}, ..., A_k \subseteq B_{il}$. The containment relationship between sequences is denoted as $s_a \sqsubseteq s_b$.

*Example 2* The sequence $\langle(a, b, c), (b, e)\rangle$ is a subsequence of the sequence illustrated in Fig. 1.

The traditional model for mining periodic patterns in a discrete sequence relies on the support and maximum periodicity measure to identify periodic patterns [4, 29].

**Definition 3 (Support of an itemset in a sequence)** Let there be an itemset $X$ and a discrete sequence $s$. For an itemset $X$, the notation $TR(X, s)$ denotes the ordered set of transactions $TR(X, s) = \langle T_{g_1}, T_{g_2}, ..., T_{g_k}\rangle \sqsubseteq s$ where $X$ appears in the sequence $s$. The support of $X$ in $s$ is the number of transactions containing $X$, that is $sup(X, s) = |TR(X, s)|$.

*Example 3* The itemset $\{a, b\}$ appears three times in the sequence of Fig. 1. Hence, the support of $\{a, b\}$ in that sequence is 3, which is denoted as $sup(\{a, b\}) = 3$.

**Definition 4 (Maximum periodicity of an itemset in a sequence)** Let there be a discrete sequence $s$. Two transactions $T_x$ and $T_y$ in $s$ are said to be *consecutive with respect to* $X$ if there does not exist a transaction $T_z \in s$ such that $x < z < y$ and $X \subseteq T_z$. The period of two consecutive transactions $T_x$ and $T_y$ for a pattern $X$ is defined as $per(T_x, T_y) = y - x$. The periods of $X$ in a sequence $s$ are $pr(X, s) = \{per_1, per_2, ..., per_{k+1}\}$ where $per_1 = g_1 - g_0$, $per_2 = g_2 - g_1, \ldots per_{k+1} = g_{k+1} - g_k$, and $g_0 = 0$ and $g_{k+1} = n$, respectively. The maximum periodicity of an itemset $X$ in a sequence $s$ is defined as $maxPr(X, s) = argmax(pr(X, s))$ [4].

*Example 4* For example, consider the itemset $X = \{a, b\}$. That itemset appears in three transactions of the sequence of Fig. 1, namely $T_1$, $T_3$ and $T_5$. Hence, it is said that $TR(X, s_1) = \{T_1, T_3, T_5\}$. The periods of $X$ in that sequence are $pr(X, s) = \{1, 2, 2, 3\}$. The maximum periodicity of $X$ in that sequence is $maxPr(X, s) = argmax(\{1, 2, 2, 3\}) = 3$.

In the traditional periodic pattern mining model, the goal is to find all the periodic itemsets in a sequence $s$. Let there be two user-specified thresholds called the maximum periodicity threshold $maxPer$ and the minimum support threshold $minSup$. An itemset $X$ is deemed periodic in a sequence $s$ if $maxPr(X, s) \leq maxPer$ and $sup(X, s) \geq minSup$.

*Example 5* If $maxPr = 3$ and $minSup = 3$, the itemset $\{a, b\}$ is periodic in the sequence of Fig. 1 because its periods are $pr(\{a, b\}, s_1) = \{1, 2, 2, 3\}$, its maximum period is $maxPr(\{a, b\}, s) = max\{1, 2, 2, 3\} = 3 \leq maxPr$ and $sup(\{a, b\}, s) = 3 \geq minSup$.

In the above example, the sequence does not contain timestamps. But it should be noted that it is simple to extend the previous model to use timestamps. This can be done by replacing transaction identifiers by timestamps.

Several algorithms have been designed for the traditional problem of mining periodic patterns in a sequence [4, 29]. Moreover, in the last decades, several variations of that problem have been proposed using alternative functions to evaluate the periodicity of each itemset $X$. Some functions that have been used to provide more flexibility are the minimum periodicity [29], average periodicity [29], and standard deviation of periods [19, 49, 50] respectively defined as $minPr(X, s) = argmin(pr(X, s))$, $avgPr(X, s) = average(pr(X, s))$ and $stanDev(X, s) = \sqrt{\frac{\sum_{v \in pr(X,s)} (v - avgPr(X, s))^2}{avgPr(X, s)}}$.

*Example 6* Continuing the previous example, $minPr(X, s_1) = argmin(\{1, 2, 2, 3\}) = 1$, $avgPr(X, s_1) = average(\{1, 2, 2, 3\}) = \frac{1+2+2+3}{4} = 2$. Also, $stanDev(X, s_1) = \sqrt{\frac{(1-2)^2+(2-2)^2+(2-2)^2+(3-2)^2}{4}} = \sqrt{0.5} \approx 0.71$.

Another function is the variance of periods [42, 53, 54]. In some other recent studies, a function to evaluate the periodic stability was introduced to find patterns that have a stable periodic behavior over time [26, 30]. The concept of periodicity has also been combined with various other functions to evaluate other aspects besides periodicity such as the utility (e.g. profit or importance) [12, 13, 22] and a statistical test [49]. However, all the above models are designed to find patterns in a single sequence rather than finding patterns that are common to multiple sequences. The next section discusses how periodic patterns can be discovered in multiple sequences.

## 3 Finding Periodic Patterns in Multiple Discrete Sequences

For many applications, the data is represented as a set of discrete sequences rather than a single sequence. This data format is called a *sequence database*, and is defined as follows.

**Definition 5 (Sequence database)** A sequence database $D$ is a set of $n$ sequences that is ordered, denoted as $D = \langle s_1, s_2, ..., s_n \rangle$. The $i$-th sequence of $D$ is denoted as $s_i$, and its sequence identifier is said to be $i$.

*Example 7* A small sequence database is shown in Table 1. This database contains four sequences that have the sequence identifiers 1, 2, 3 and 4. These four sequences could indicate the sequence of purchases made by four customers in a store. For example, sequence 1 indicates that a customer purchased items $a$, $b$ and $e$ together, followed by $a$, $b$ and $e$, followed by $a$ and $d$, followed by $a$ and $e$, and lastly followed by $a$, $b$ and $c$.

Table 1: An example sequence database

| Identifier | Sequence |
|---|---|
| 1 | $\langle \{a, b, e\}, \{a, b, e\}, \{a, d\}, \{a, e\}, \{a, b, c\} \rangle$ |
| 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 | $\langle \{a, b, d, e\}, \{a, b, e\}, \{a, b, c\}, \{a, b, d, e\}, \{a, b\} \rangle$ |

For such data, it can be desirable to find patterns that are periodic in multiple sequences rather than to consider each sequence separately. For this purpose, an evaluation function called the *sequence periodic ratio* was proposed [19, 28]. This function is, defined as follows.

**Definition 6 (Sequence periodic ratio)** The number of sequences where an itemset $X$ is periodic in a sequence database $D$ is denoted and defined as $numSeq(X)$. The sequence periodic ratio of $X$ in $D$ is defined as $ra(X) = numSeq(X)/|D|$, where $|D|$ is the number of sequences in $D$.

In other words, the sequence periodic ratio represents the percentage of sequences from a database where an itemset $X$ is periodic. Then, the problem of finding periodic patterns common to multiple sequences is defined as follows.

**Definition 7 (General problem of mining periodic patterns common to multiple sequences)** Let there be a sequence database, a minimum sequence periodic ratio threshold $minRa$ and a definition of what is a periodic pattern in a single sequence. An itemset $X$ is a periodic patterns in $D$ if $ra(X) \geq minRa$. The problem of *mining periodic patterns common to multiple sequences* is to find all periodic patterns in $D$.

It is to be noted that the above problem definition and sequence periodic ratio are defined in a general way such that various pattern evaluation functions could be used to evaluate if a pattern is periodic in each sequence. In the following two subsections, two instanciations of the general problem are presented to respectively find (1) periodic frequent patterns in multiple sequences and (2) rare correlated periodic patterns. For each of these problem instantiation, different evaluation functions are used to determine if a pattern is periodic in a sequence.

### 3.1 Finding Frequent Periodic Patterns

The first problem instanciation is designed to find periodic patterns that are frequent, that is that appear in many sequences of a sequence database. For example, an application is to find periodic patterns that are common to many customers of a retail store. This problem is defined as follows [19].

**Definition 8 (Problem of mining periodic frequent patterns common to multiple sequences)** Let there be a sequence database $D$, and four user-defined thresholds, namely the minimum support threshold $minSup$, maximum periodicity threshold $maxPr$, maximum standard deviation threshold $maxStd$, and minimum sequence periodic ratio threshold $minRa$. In that definition, an itemset $X$ is periodic in a sequence $s$ if $maxPer(X,s) \leq maxPr \wedge sup(X,s) \geq minSup \wedge stanDev(X,s) \leq maxStd\}|$. The *problem of mining frequent periodic patterns common to multiple sequences* is to find all Periodic Frequent Patterns (PFP) [19]. An itemset $X$ is a PFP in $D$ if $ra(X) \geq minRa$.

*Example 8* Consider the database $D$ of Table 1, and that the user specifies some thresholds $minSup = 2$, $maxPr = 3$, $maxStd = 5.0$ and $minRa = 0.3$. Let $X = \{a, b\}$. The itemset $X$ is periodic in sequence $s_1$, $s_2$ and $s_4$. Thus, $numSeq(X) = 3$. Furthermore, as the database contains four sequences, i.e. $|D| = 4$, the sequence periodic ratio of $X$ is $ra(X) = 3/4 = 0.75$. Hence, $X$ is a PFP in the database. The list of all PFP that can be found for different threshold values are shown in Table 2.

Table 2: Periodic Frequent Patterns found in the database of Fig. 1 for different threshold values

| $minSup$ | $maxPr$ | $maxStd$ | $minRa$ | Periodic frequent patterns found |
|---|---|---|---|---|
| 2 | 3 | 1.0 | 0.6 | $\{a\}, \{e\}, \{a, e\}$ |
| 2 | 3 | 1.0 | **0.4** | $\{a\}, \{b\}, \{c\}, \{e\}, \{a, b\}, \{a, e\}, \{b, e\}, \{a, b, e\}$ |
| 2 | **1** | 1.0 | 0.6 | $\{a\}$ |
| **3** | 3 | 1.0 | 0.6 | $\{a\}$ |
| 2 | 3 | **1.5** | 0.6 | $\{a\}, \{b\}, \{e\}, \{a, b\}, \{a, e\},$ |

It is interesting to observe that the traditional problem of mining periodic frequent patterns in a single sequence can be viewed as a special case of the general problem of mining periodic frequent patterns common to multiple sequences such that the sequence database contains a single sequence and $minRa = 0$.

### 3.2 Finding Rare Correlated Periodic Patterns

The previous problem instantiation is designed to find periodic patterns that are frequent (appear in many sequences). Although these patterns may be interesting, rare patterns can also be valuable. To be able to find rare patterns, a second instantiation of the general problem of mining periodic patterns common to multiple sequence was defined to find rare correlated periodic patterns [28]. The motivation is to find patterns that are periodic, appear rarely but contain items that are strongly correlated in a sequence database. For the example of analyzing customer data, this can mean to find patterns that are periodic and rare (not purchased by many customers) but have a strong correlation.

To find rare patterns in data, several methods have been proposed in the literature [41, 45, 58, 59] using different definitions of what is a rare pattern and how to efficiently find these patterns. For an overview of studies on rare pattern mining, the reader can refer to a recent survey about finding rare patterns by Koh et al. [40]. In the following, a simple definition of what is a rare pattern is used. It is that an *itemset is rare* if its support is not greater than some maximum support threshold $maxSup$, set by the user.

But a problem with rare patterns is that because they seldomly appear, it is possible that some of them contains items that appear together by chance. To avoid this problem and find patterns representing items that are strongly correlated, several correlation measures have been used in itemset mining such as the bond [20, 32, 51, 68], affinity [3], all-confidence [51, 62], coherence and mean [7, 56], each having different advantages and limitations [33]. In the problem instantiation presented in this section, the bond measure is used because it is a simple measure, it is easy to calculate and also easy to interpret. The bond of an itemset is defined based on the following definitions.

**Definition 9 (Disjunctive support)** Let there be a sequence $s$ and an itemset $X$. The disjunctive support of $X$ in $s$ is the number of transactions that contain one or more items from $X$. It is denoted as $dissup(X, s)$ and defined as $dissup(X, s) = max\{sup(X, s)|X \in s\}$

**Definition 10 (Bond)** Let there be a sequence $s$ and an itemset $X$. The bond of $X$ in $s$ is defined as $bond(X, s) = \frac{sup(X,s)}{dissup(X,s)}$.

The bond of an itemset can take a value in the $[0, 1]$ interval such that a value of 0 indicates a low correlation and a value of 1 indicates the maximum correlation.

*Example 9* The disjunctive support of the itemset $\{a, e\}$ in the first sequence ($s_1$) of the database of Table 3 is calculated as $dissup(\{a, e\}, s_1) = 5$, while the support of $\{a, e\}$ is $sup(\{a, e\}) = 3$. Hence, the bond of that itemset is $bond(\{a, e\}, s_1) = \frac{3}{5} = 0.6$.

Table 3: A second example 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 | $\langle\{a, b, d, e\}, \{a, b\}, \{a, c\}, \{a, b, d, e\}, \{a, d\}\rangle$ |

Based on the concept of rare patterns and the bond measure for identifying correlated patterns, the second problem instantiation is defined.

**Definition 11 (Rare correlated periodic pattern in a sequence)** Let there be three thresholds, namely the maximum support threshold ($maxSup$), minimum bond threshold ($minBond$) and maximum standard deviation threshold ($maxStd$). An itemset $X$ is a rare correlated periodic pattern in a sequence $s$ if $sup(X, s) \leq maxSup$, $stanDev(X, s) \leq maxStd$ and $bond(X, s) \geq minBond$.

*Example 10* The periods of itemset $X = \{a\}$ in the sequence $s_2$ are $pr(X, s_2) = \{2, 2, 1, 0\}$. Hence, $sup(X, s_2) = 4$, $dissup(X, s_2) = 4$, $bond(\{a\}) = \frac{4}{4} = 1$, $maxPr(X, s_2) = 2$ and $avgPr(X, s_2) = \frac{2+2+1+0}{4} = 1.25$. The standard deviation of periods is $stanDev(X, s_2) = \sqrt{\frac{(2-1.25)^2+(2-1.25)^2+(1-1.25)^2+(0-1.25)^2}{4}} = \sqrt{0.6875} \approx 0.83$. Thus, if $minBond = 0.5$, $maxStd = 0.9$ and $maxSup = 4$, the itemset $X$ is a rare correlated periodic pattern in sequence $s_2$.

**Definition 12 (Sequence periodic ratio)** In the context of discovering rare correlated periodic patterns, the number of sequences where an itemset $X$ is a rare correlated periodic pattern in a sequence database $D$ is denoted and defined as $numSeq(X)$. Moreover, the sequence periodic ratio of $X$ in $D$ is defined as $ra(X) = numSeq(X)/|D|$, where $|D|$ is the number of sequences in $D$.

**Definition 13 (Problem of mining periodic rare correlated periodic patterns common to multiple sequences)** Consider a sequence database $D$ and four user-specified thresholds: a maximum support threshold ($maxSup$), a minimum bond threshold ($minBond$), a maximum standard deviation threshold ($maxStd$) and a minimum sequence periodic ratio threshold ($minRa$). The problem of mining RCPP common to multiple sequences is to find all the Rare Correlated Periodic Patterns (RCPP) in the database $D$, that is each itemset $X$ where $ra(X) \geq minRa$ [28].

*Example 11* Consider the database of Table 3 and that the thresholds are set by the user as $maxSup = 2$, $maxStd = 1$, $minBond = 0.6$ and $minRa = 0.70$. The itemset $\{b, e\}$ is a rare correlated periodic pattern in sequence $s_1$, $sup(\{b, e\}, s_1) = 2$, $stanDev(\{b, e\}, s_1) = 0.47$ and $bond(\{b, e\}, s_1) = 0.67$. Moreover, it can be found that $\{b, e\}$ is also a rare correlated periodic pattern in sequence $s_2$ and $s_4$. Hence, $numSeq(\{b, e\}) = 3$ and the sequence periodic ratio of {b,e} is $ra(\{b, e\}) = 3/4 = 0.75$. Thus, $\{b, e\}$ is a RCPP.

The problem of discovering RCPP in a database is hard because the maximum support and standard deviation functions cannot be directly used to reduce the search space, as these functions are neither monotonic nor anti-monotonic. Thus, other ideas must be used to design efficient algorithms. Those will be discussed in the next section describing algorithms.

## 4 Two Algorithms

This section presents two efficient algorithms for mining periodic patterns in multiple sequence. The first subsection presents the MPFPS (Mining Periodic Frequent Patterns in multiple Sequences) algorithm to solve the first problem instantiation from Section 3.1, while the following subsection presents the MRCPPS algorithm to solve the problem instantiation of Section 3.2.

### 4.1 The MPFPS Algorithm to Mine for Mining Frequent Periodic Patterns in Multiple Sequences

The key challenge to find periodic patterns in multiple sequences is how to reduce the search space to avoid looking at all the possible itemsets. In the field of pattern mining, several properties of pattern evaluation functions are used to reduce the search space. For the problem of finding periodic frequent patterns in multiple sequences, an upper bound on the sequence periodic ratio is used in the MPFPS algorithm for reducing the search space [19, 28], which is defined as follows.

**Definition 14** Let there be an itemset $X$, a sequence $s$ and the $maxPer$ and $minSup$ thresholds. The itemset $X$ is a *candidate* in a sequence $s$ if $sup(X, s) \geq minSup$ and $maxPr(X, s) \leq maxPr$. The number of sequences where an itemset $X$ is a candidate is denoted as $numCand(X)$ and defined as $numCand(X) = |\{s | maxPr(X, s) \leq maxPr \wedge sup(X, s) \geq minSup \wedge s \in D\}|$. The *boundRa* upper bound of $X$ is denoted as $boundRa(X)$ and defined as $boundRa(X) = numCand(X)/|D|$.

Two properties of the *boundRa* upper bound are utilized by MPFPS to reduce the search space.

**Theorem 1** *For any itemset $X$, the relationship $boundRa(X) \geq ra(X)$ holds* [19].

***Proof*** We have an itemset $X$ and:

$$\begin{aligned} ra(X) &= numSeq(X)/|D| \\ &= |\{s|maxPr(X,s) \leq maxPr \wedge sup(X,s) \\ &\quad \geq minSup \wedge stanDev(X,s) \leq maxStd \wedge s \in D\}|/|D| \\ &\leq |\{s|maxPr(X,s) \leq maxPr \wedge sup(X,s) \geq minSup \wedge s \in D\}|/|D| \\ &= numCand(X)/|D| \\ &= boundRa(X) \end{aligned}$$

**Theorem 2** *Let there be two itemsets $X$ and $X'$ such that $X' \subseteq X$. Then, $boundRa(X') \geq boundRa(X)$*[19].

**Theorem 3** *If $boundRa(X') < minRa$ for an itemset $X'$, then $X'$ and any superset $X \supset X'$ are not PFP* [19].

The proof of the above theorems can be found in the paper presenting the MPFPS algorithm [19].

The MPFPS algorithm searches for periodic patterns by first starting from 1-itemsets, and then it recursively adds an item to each itemset to search for larger itemsets. The items are appended to an itemset by following a total order on item called $\succ$. During the search for periodic itemsets, the MPFPS relies on the above theorem to reduce the search space. If an itemset $X$ has a *boundRa* that is smaller than the *minRa* threshold set by the user, all its supersets cannot be PFP and thus can be ignored to reduce the search space.

To be able to decide if an itemset is periodic, it is necessary to be able to calculate its support, periods and its sequence periodic ratio. This is done using a special data structure named a PFPS-list. This data structure is constructed for each itemset that is considered by the algorithm during the search for PFP. It is defined based on the following definitions.

**Definition 15 (Sequences containing an itemset)** The ordered list of sequence containing an itemset $X$ is denoted as *sequences*$(X)$ and defined as a list *sequences*$(X) = \{s|sup(X,s) > 0 \wedge s \in D\}$, ordered by increasing sequence identifiers.

*Example 12* For Table 1, the list of sequences containing $\{e\}$ is *sequences*$(\{e\})$ $= \{s_0, s_1, s_3\}$.

**Definition 16 (The PFPS-list structure)** Let there be an itemset $X$ and a sequence database $D$. The PFPS-list $LX$ of $X$ is a table containing three rows (fields) called *i-set*, *sid-list* and *tidlist-list*. The first row contains the itemset $X$, that is $LX.i\text{-}set = X$. The second row stores the list of identifiers of sequences containing $X$, that is $LX.sidlist = \{v_1, v_2 \ldots v_w\}$ where $w = |sequences(X)|$ and $v_i$ $(1 \leq i \leq w)$ is the sequence identifier of the $i$-th sequence in $sequences(X)$. The third row, *tidlist-list* contains the identifiers of transactions containing $X$, for each sequence containing $X$. It is formally defined as a list $LX.tidlist\text{-}list = \{Z_1, Z_2 \ldots Z_w\}$ such that $w = |sequences(X)|$. Let $s_i$ be the $i$-th sequence in $sequences(X)$ $(1 \leq i \leq w)$. The value $Z_i$ is $Z_i = \{z | X \subseteq T_z \wedge \langle T_z \rangle \sqsubseteq s_i\}$.

*Example 13* The PFPS-lists of itemsets $\{a\}$, $\{e\}$ and $\{a, e\}$ are presented in Figure 2 for the database of Table 1.

| *i-set* | $\{a\}$ |
|---|---|
| *sid-list* | {0,1,2,3} |
| *tidlist-list* | [{0,1,2,3,4}, {1,3,4},{1,2,3,4}, {0,1,2,3,4}] |

| *i-set* | $\{e\}$ |
|---|---|
| *sid-list* | {0,1,3} |
| *tidlist-list* | [{0,1,3}, {1,3,4}, {0,1,3}] |

| *i-set* | $\{a, e\}$ |
|---|---|
| *sid-list* | {0,1,3} |
| *tidlist-list* | [{0,1,3}, {1,3}, {0,1,3}] |

Fig. 2: The PFPS-lists of itemsets $\{a\}$ (top-left), $\{e\}$ (top-right) and $\{a, e\}$ (bottom)

The PFPS-list of an itemset $X$ indicates (1) the list of sequences where $X$ appears (in the field *sid-list*), and (2) the list of transactions containing $X$ for any sequence $s$ (in the field *tidlist-list*). This latter information is all is needed to calculate $pr(X, s)$, to then derive $sup(X, s)$, $maxpr(X, s)$ and $stanDev(X, s)$, and thus check if $X$ is a periodic frequent pattern. Thus, using the PFPS-list structure, it is possible to check if an itemset is periodic without scanning the database.

The MPFPS algorithm initially reads the database once to build the PFPS-list of each item. Then, MPFPS starts to search for larger itemsets. During this search, the PFPS-lists of each larger itemset is obtained by joining the PFPS-lists of two of its subsets. Thus, building the PFPS-lists of itemsets having more than 1 item does not require scanning the database again.

For an itemset $P$ and some items $x$ and $y$, the itemset $P \cup \{x\}$ is said to be an *extension* of $P$, which is briefly denoted as $Px$. The algorithm for constructing the PFPS-list of an itemset $Pxy$ by joining the PFPS-lists of two itemsets $Px$ and $Py$ is shown in Algorithm 1 [19]. For instance, this algorithm can be applied to the PFPS-lists of $\{a\}$ and $\{e\}$ to build the PFPS-list of $\{a, e\}$, which are all depicted in Fig 2.

The MPFPS (Algorithm 2) takes as input a sequence database $D$ and the user-defined thresholds $maxStd$, $minRa$, $maxPr$, and $minSup$. The algorithm explores the search space using a depth-first search, and returns the set of all PFP. MPFPS first reads the input database to calculate the following information for each item $i$ and sequence $s$: $sup(\{i\}, s)$, $pr(\{i\}, s)$, $maxpr(\{i\}, s)$ and $stanDev(\{i\}, s)$. A loop is then done by MPFPS on each single item. An

**Algorithm 1:** The Intersect procedure

```
input  : the PFPS-lists $LPx$ and $LPy$ of some itemsets $Px$ and $Py$
output: the PFPS-list $LPxy$ of itemset $Pxy$
$LPxy.i\text{-}set \leftarrow Px \cup \{y\}$; $LPxy.tidlist\text{-}list \leftarrow \emptyset$; $LPxy.sid\text{-}list \leftarrow \emptyset$;
foreach sequence identifier $sid \in LPx.sid\text{-}list$ such that $sid \in LPy.sid\text{-}list$ do
    $tidListSidPx \leftarrow$ the tid list of $sid$ in $LPx.tidlist\text{-}list$;
    $tidListSidPy \leftarrow$ the tid list of $sid$ in $LPy.tidlist\text{-}list$;
    $tidListSidPxy \leftarrow tidListSiPx \cap tidListSiPy$;
    if $tidListSidPxy \neq \emptyset$ then
        $LPxy.sid\text{-}list.append(sid)$; $LPxy.tidlist\text{-}list.append(tidListSidPxy)$;
    end
end
return $LPxy$;
```

item $i$ is considered to be periodic in a sequence $s$ if $sup(\{i\}, s) \geq minSup$, $maxpr(\{i\}, s) \leq maxPr$ and $stanDev(\{i\}, s) < maxStd$. Then, the algorithm calculates the sequence periodic ratio of item $i$ by dividing the number of sequences where $i$ is periodic by the total number of sequences (line 3 to 4). If this value is not less than $minRa$, $i$ is a PFP and it is output (line 5). Thereafter, the algorithm calculates the $boundRa$ upper bound of $\{i\}$ (line 6 to 7). After the loop is completed, for each item $i$ such that $boundRa(\{i\}) \geq minRa$, the PFPS-list of $\{i\}$ is put in a list $boundPFPS$ (line 9), which is then sorted by the total order $\succ$ of increasing $boundRa$ values. After that, a $Search$ procedure is invoked to recursively search for extensions of single items in $boundPFPS$. Other itemsets do not need to be explored according to Theorem 3. The $Search$ procedure is initially called with $boundPFPS$, $minSup$, $maxPr$, $maxStd$, $minRa$ and $D$.

The $Search$ procedure is described in Algorithm 3. The input of the procedure is the PFPS-lists of extensions of an itemset $P$. Recall that an extension of an itemset $P$ with an item $z$ refers to the itemset obtained by appending an item $z$ to $P$, and is denoted as $Pz$. Moreover, the four thresholds are also received as parameters, as well as the input database $D$. When the $Search$ procedure is first called, $P = \emptyset$ and all its extensions are 1-itemsets (items). The procedure executes two loops to join all pairs of extensions of $P$, denoted as $Px$ and $Py$ where $x, y$ are items and $y \succ x$, (line 1 to 12). The result of a join is an extension $Pxy$ containing $|Px| + 1$ items. The PFPS-list of $Pxy$ is obtained by calling the Algorithm 1 with the PFPS-lists of $Px$ and $Py$ as parameters (line 3). Then, the procedure calculates $boundRa(Pxy)$ and $numCand(Pxy)$ by reading $Pxy$ (line 4 to 5). In the case, where the condition $boundRa(Pxy) \geq minRa$ is not met, $Pxy$ and all its supersets are not PFP and can be ignored from further processing according to Theorem 3 (line 6). If the condition is met, the PFPS-list of $Pxy$ is stored into a variable $ExtensionsOfPx$ that contains all PFPS-lists of extensions of $Px$ having a $boundRa$ value that is no less than $minRa$ (line 7). Thereafter, the algorithm calculates the ratio $ra(Pxy)$ and if $ra(Pxy) \geq minRa$, then $Pxy$ is output as a PFP (line 8 to 10). Finally, a recursive call of the $Search$ procedure is made with $ExtensionsOfPx$ (PFPS-lists of itemsets extending $Px$ that are PFP) to search for potential transitive extensions that are also PFP (line 13).

When the algorithm terminates, all PFP have been output. The proof that the algorithm is complete is ommitted but it can be easily seen based on Theorem 3, as it guarantees that the algorithm only ignores itemsets that are non PFP.

**Algorithm 2:** The MPFPS algorithm

```
input  : $D$: a sequence database, $maxStd, minRa, maxPr, minSup$: the
         thresholds.
output: the set of periodic frequent patterns (PFPS).
Scan each sequence $s \in D$ to calculate $sup(\{i\}, s)$, $pr(\{i\}, s)$, $maxpr(\{i\}, s)$
  and $stanDev(\{i\}, s)$ for each item $i \in I$;
foreach item $i \in I$ do
    $numSeq(\{i\}) \leftarrow |\{s | maxpr(\{i\}, s) \leq maxPr \wedge stanDev(\{i\}, s) \leq$
      $maxStd \wedge sup(\{i\}, s) \geq minSup \wedge s \in D\}|$;
    $ra(\{i\}) \leftarrow numSeq(\{i\})/|D|$;
    if $ra(Px) \geq minRa$ then output $Px$;
    $numCand(\{i\}) \leftarrow |\{s | maxpr(\{i\}, s) \leq maxPr \wedge sup(\{i\}, s) \geq$
      $minSup \wedge s \in D\}|$;
    $boundRa(\{i\}) \leftarrow numCand(\{i\})/|D|$;
end
$boundPFPS \leftarrow \{$PFPS-list of item $i | i \in I \wedge boundRa(\{i\}) \geq minRa\}$;
Sort $boundPFPS$ by the order $\succ$ of ascending $boundRa$ values;
Search ($boundPFPS$, $minSup$, $maxPr$, $maxStd$, $minRa$, $D$);
```

To illustrate how the algorithm is applied, a detailled example is next presented for mining PFP in the sequence of Table 1. Consider that the parameters are set as $minRa = 0.6$, $maxPr = 3$, $minSup = 2$, and $maxStd = 1.0$. The following steps are performed.

1. The algorithm reads the database and first checks single items. Consider the item $a$. The periods, maximum periodicity and standard deviation of $a$ are calculated for each sequence. The obtained values are shown in Table 4. Based on these values, it is found that $a$ is periodic in the four sequences. For instance, $a$ is periodic in $s_2$ because $maxpr(\{a\}, s_2) = 2 < maxPr$ and $stanDev(\{a\}, s_2) = 0.256 < maxStd$. The number of sequence where $a$ is periodic is $numSeq(\{a\}) = 4$ and as there are four sequences in the database, $ra(\{a\}) = 4/4 = 1 \geq minRa$. Hence, $\{a\}$ is output as a PFP. The same process is repeated for all other items. It is found that only the items $\{a\}$ and $\{e\}$ are PFP.
2. Then, for each item having a $boundRa$ value that is greater or equal to $minRa$, its PFPS-list is created and added to $boundPFPS$. In this exam-

**Algorithm 3:** The *Search* procedure

```
input  : $ExtensionsOfP$: a set of PFPS-lists of extensions of an itemset $P$,
         $minSup, maxPr, maxStd, minRa$: the thresholds, $D$: the database.
output: the set of periodic frequent patterns that extend $P$.
foreach $PFPS\text{-}list\ LPx \in ExtensionsOfP\ and\ Px = LPx.i\text{-}set$ do
    foreach $PFPS\text{-}list\ LPy \in ExtensionsOfP\ and\ Py = LPy.i\text{-}set\ such\ that\ y \succ x$ do
        $LPxy \leftarrow$ Intersect $(LPx, LPy)$;
        $numCand(Pxy) \leftarrow |\{s | maxpr(Pxy, s) \leq maxPr \wedge sup(Pxy, s) \geq minSup \wedge s \in D\}|$;
        $boundRa(Pxy) \leftarrow numCand(Pxy)/|D|$;
        if $boundRa(Pxy) \geq minRa$ then
            $ExtensionsOfPx \leftarrow ExtensionsOfPx \cup \{LPxy\}$;
            $numSeq(Pxy) \leftarrow |\{s | maxpr(Pxy, s) \leq maxPr \wedge stanDev(Pxy, s) \leq maxStd \wedge sup(Pxy, s) \geq minSup \wedge s \in LPxy.sid\text{-}list\}|$;
            $ra(Pxy) \leftarrow numSeq(Pxy)/|D|$;
            if $ra(Pxy) \geq minRa$ then output $Pxy$;
        end
    end
    Search $(ExtensionsOfPx, minSup, maxPr, maxStd, minRa, D)$;
end
```

ple, only the PFPS-lists of $\{a\}$ and $\{e\}$ are built. The *boundRa* value of $\{a\}$ is 1.

3. After this, the list *boundPFPS* is sorted by increasing *boundRa* values and the algorithm invokes the *Search* procedure to find extensions of $\{a\}$ and $\{e\}$ that are PFP (Algorithm 3).
4. The *Search* procedure is a recursive method that extends itemsets having their PFPS-lists in *boundPFPS* by joining pairs of them. Initially, all PFPS-lists in *boundPFPS* are extensions of the empty set and the aim is to generate itemsets having two items.
5. The procedure first considers combining $\{a\}$ and $\{e\}$ to obtain $\{a, e\}$. These two itemsets can be joined because they have a common prefix (the empty set). To create the PFPS-list of $\{a, e\}$, the *Intersect* method is called with the PFPS-lists of $\{a\}$ and $\{e\}$ as parameters. The *sid-list* of $\{a, e\}$ is obtained by calculating the union of the *sid-lists* of $\{a\}$ and that of $\{e\}$ as $\{0, 1, 2, 3\} \bigcup \{0, 1, 3\} = \{0, 1, 3\}$. For sequence $s_1$, the tidlist of $\{a\}$ is $\{0, 1, 2, 3, 4\}$ while that of $\{e\}$ is $\{0, 1, 3\}$. Their intersection is $\{0, 1, 2, 3, 4\} \bigcap \{0, 1, 3\} = \{0, 1, 3\}$. Hence, the sequence identifier 1 is added to *sid-list* of $\{a, e\}$ and $\{0, 1, 3\}$ is stored in the *tidlist-list* of $\{a, e\}$. The same process is repeated to build the remaining part of the PFPS-list of $\{a, e\}$. The final PFPS-list of $\{a, e\}$ is shown at the bottom of Fig. 2.
6. Based on the PFPS-list of $\{a, e\}$, it is calculated that $boundRa(\{a, e\}) = 0.6$ which is no less than *minRa*. Hence, $\{a, e\}$ may be a PFP as well as its supersets.

7. The next step is to check in which sequences $\{a, e\}$ is periodic to obtain $ra(\{a, e\})$. As $numSeq(\{a, e\}) = 3$, we have $ra(\{a, e\}) = 0.75 \geq minRa$. Hence, the itemset $\{a, e\}$ is output as a PFP.
8. The *Search* procedure then applies this process again to generate other itemsets and recursively calls itself to find their extensions, in the same way.

Table 4: Periods, maximum periodicity and standard deviation of itemset $\{a\}$ in sequences from Table 1.

| Sequence $s_x$ | $pr(a, s_x)$ | $maxPr(a, s_x)$ | $stanDev(a, s_x)$ | {a} is periodic in $s_x$? |
|---|---|---|---|---|
| $s_1$ | [1,1,1,1,1,0] | 1 | 0.152 | yes |
| $s_2$ | [2,2,1,0] | 2 | 0.256 | yes |
| $s_3$ | [2,1,1,1,0] | 2 | 0.632 | yes |
| $s_4$ | [1,1,1,1,1,0] | 1 | 0.152 | yes |

It is to be noted that the MPFPS algorithm described in that subsection was originally called $MPFPS_{depth}$ [19]. There also exists a breadth-first search version of this algorithm called $MPFPS_{breadth}$ [19], which is not presented in this chapter due to space limitation. In terms of performance, both $MPFPS_{depth}$ and $MPFPS_{breadth}$ perform better on some datasets.

The MPFPS algorithm was applied to analyze text data where each sequences of a book can be viewed as a sequence. Then, the discovered PFP can reveal patterns that represents the writing styles of authors. The algorithm was also applied to find patterns in sequences of clicks on the FIFA website, where it was discovered that some webpages are periodically clicked by several users [19].

### 4.2 The MRCPPS algorithm

The MRCPPS algorithm [28] was created by adapting the MPFPS algorithm to solve the second problem instantiation described in this chapter. Since both algorithms are designed to find periodic patterns in multiple sequences, they have many similarities. The key differences lie in the fact that these algorithms use different functions to select patterns. While MPFPS aims to find frequent periodic patterns, MRCPPS is designed to find rare correlated patterns using the bond measure.

The MRCPPS algorithm applies an upper bound on the *bond* measure to reduce the seach space, which is called *upBondRa*, which is defined below.

**Definition 17 (upBondRa)** Let there be an itemset $X$. It is considered to be a candidate in a sequence $s$ if $bond(X, s) \geq minBond$, where $minBond$ is a user-defined threshold. Then, the number of sequences where an itemset

$X$ is a candidate in a sequence database $D$ is denoted as $numCand(X)$. The *upBondRa* of X in $D$ is defined as $upBondRa(X) = numCand(X)/|D|$.

The following theorem indicates that *upBondRa* can be used to reduce the search space [28].

**Theorem 4** *Let there be two itemsets $X \subset X'$. Then, the two following relationships hold: $upBondRa(X) \geq ra(X)$ and $upBondRa(X) \geq upBondRa(X')$.*

To calculate the *bond*, $maxPr$ and other evaluation functions for an itemset, the MRCPPS algorithm utilizes a novel data structure called RCPPS-list. This structure can be viewed as a more complex structure than the PFPS-list presented in the previous subsection. The reason is that a RCPPS-list stores additional information required to calculate the bond measure.

A RCPPS-list is created for each itemset $X$ considered by MRCPPS. The RCPPS-list of $X$ is a table providing information about an itemset $X$ stored in three rows (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.

*Example 14* The RCPPS-lists of $\{b\}$, $\{e\}$ and $\{b, e\}$ are shown in Fig. 3, for the database of Table 3.

Several useful information can be calculated directly from the RCPPS-list of an itemset. The field SIDlist provides the list of sequences containing the itemset. For instance, the field *SIDlist* of the RCPPS-list of $\{b, e\}$ indicates that it appears in sequences $s_1$, $s_2$ and $s_4$. The size of *conTIDlist* for a sequence indicates the support of the itemset in that sequence, while the *bond* can be calculated as the size of its *conTIDlist* divided by the size of its *disTIDlist* for the sequence. The *conTIDlist* and *disTIDlist* of itemset $\{b, e\}$ for sequence $s_1$ are [2, 4] and [1, 2, 4], respectively. As a result, the bond of $\{b, e\}$ in sequence $s_1$ is $\frac{2}{3}$.

The *stanDev* and periods of an itemset can also be calculated from its RCPPS-list, as well as the *upBondRa* and *ra* values. The *ra* is used by the RCPP algorithm to determine if an itemset is a RCPP and the *upBondRa* value is utilized to check if it should be extended to find larger RCPP based on Theorem 4.

The procedure *Construct* for intersecting two RCPPS-lists of some itemsets $Px$ and $Py$ to obtain the RCPPS-list of an itemset $Pxy$ is shown in Algorithm 4. As the principle of that procedure is similar to the *Intersect* procedure of MPFPS, it is not described in more details here.

The pseudo-code of MRCPPS is shown in Algorithm 5. The input is the thresholds $maxSup$, $maxStd$, $minBond$ and $minRa$, and a sequence database $D$. MRCPPS initially reads the sequence database to construct the RCPPS-lists of all items. Thereafter, the algorithm creates a set $I^*$ to store

| *i-set* | $\{b\}$ |
|---|---|
| *sid-list* | {1,2,3,4} |
| *list-conTIDlist* | {[2, 4], [2, 4, 5], [1, 2, 5], [1, 2, 4]} |
| *list-disTIDlist* | {[2, 4], [2, 4, 5], [1, 2, 5], [1, 2, 4]} |

| *i-set* | $\{e\}$ |
|---|---|
| *sid-list* | {1,2,4} |
| *list-conTIDlist* | {[1,2, 4], [2, 4], [1, 4]} |
| *list-disTIDlist* | {[2, 4], [2, 4], [1, 4]} |

| *i-set* | $\{b, e\}$ |
|---|---|
| *sid-list* | {1,2,4} |
| *list-conTIDlist* | {[2, 4], [2, 4], [1, 4]} |
| *list-disTIDlist* | {[2, 4], [2, 4], [1, 4]} |

Fig. 3: The RCPPS-lists of itemsets $\{b\}$ (top-left), $\{e\}$ (top-right), and $\{b, e\}$ (bottom) for the database of Table 3

**Algorithm 4:** The Construct procedure

```
input: $LPx$: the RCPPS-list of $Px$, $LPy$: the RCPPS-list of $Py$.
out  : the RCPPS-list of $Pxy$
$LPxy \leftarrow \emptyset$;
foreach $i, j$ where $LPx.SIDlist(i) = LPy.SIDlist(j)$ do
    $conTIDlist \leftarrow LPx.list\text{-}conTIDlist(i) \cap LPy.list\text{-}conTIDlist(j)$;
    if $conTIDlist \neq \emptyset$ then
        $disTIDlist \leftarrow LPx.list\text{-}disTIDlist(i) \cup LPy.list\text{-}disTIDlist(j)$;
        $LPxy.SIDlist \leftarrow LPxy.SIDlist \cup LPx.SIDlist(i)$;
        $LPxy.list\text{-}conTIDlist \leftarrow LPxy.list\text{-}conTIDlist \cup conTIDlist$;
        $LPxy.list\text{-}disTIDlist \leftarrow LPxy.list\text{-}disTIDlist \cup disTIDlist$;
    end
end
return $LPxy$;
```

1-itemsets (single items) that have an *upBondRa* value that is greater or equal to *minRa*. Items not in $I^*$ will thereafter not be considered by the algorithm, based on Theorem 4. Then, MRCPPS sorts items in $I^*$ according to the ascending order $\succ$ of *upBondRa* values. After that, the recursive *Find* procedure (Algorithm 6) is called with the empty itemset $\emptyset$, the set of single items $I^*$, and the user-defined thresholds.

The *Find* procedure takes as input an itemset $P$, extensions of the form $Pz$ such that $upBondRa(Pz) \geq minRa$, the sequence database $D$, $maxSup$, $maxStd$, $minBond$ and $minRa$. When the *Find* procedure is first called, $ExtensionsOfP$ contains single items and $P = \emptyset$. The procedure performs a loop to process each extension $Px$ of $P$ (line 1 to 15). The value $numSeq(Px)$ and $ra(Px)$ are first calculated based on the information stored in the RCPPS-list of $Px$ (line 2 to 3). If the $ra(Px)$ value is no less than $minRa$, $X$ is output as a RCPP (line 4). And in the case where $upBondRa(Px) \geq minRa$, the procedure will consider extending $Px$ to generate larger itemsets. This is achieved by joining $Px$ with each extension of the form $Py$ where $y \succ x$ to obtain an itemset $Pxy$ (line 7). The algorithm builds $Pxy$'s RCPPS-list by calling the *Construct* procedure using the RCPPS-list of $Px$ and $Py$ (line 8). Each extension $Pxy$ is added to a set $ExtensionsOfP$ of extensions that can be considered for further extensions if the *upBondRa* value of $Pxy$ is no less than *minRa* (line 11 to 13). Extensions having an *upBondRa* value smaller than *minRa* are ignored to reduce the search space (by Theorem 4). Finally, a call to the *Search* procedure is made with $Pxy$ as argument to calculate its $ra$ value and explore its extensions(s) using a depth-first search (line 16). When the algorithm terminates all RCPP have been output.

**Algorithm 5:** The MRCPPS algorithm

```
input  : $D$: a sequence database,
         $maxSup$, $maxStd$, $minBond$, $minRa$: the user-specified threshods.
output: the set of RCPPS
Scan $D$ to calculate the RCPPS-lists of all items in $I$;
$I^* \leftarrow \emptyset$;
foreach $item\ i \in I$ do
    $numCand(i) \leftarrow |\{s|bond(i,s) \geq minBond \wedge s \in D\}|$;
    $upBondRa(i) \leftarrow numCand(i)/|D|$;
    if $upBondRa(i) \neq 0 \wedge upBondRa(i) \geq minRa$ then $I^* \leftarrow I^* \cup \{i\}$;
end
Sort $I^*$ by the order $\succ$ of ascending $upBondRa$ values;
Find ($\emptyset$, $I^*$, $D$, $minSup$, $maxStd$, $minBond$, $minRa$);
```

**Algorithm 6:** The Find procedure

```
input  : $P$: an itemset, $ExtensionsOfP$: a set of extensions of $P$, $D$: the
         sequence database
         $maxSup$, $maxStd$, $minBond$, $minRa$: the user-specified threshods.
output: the set of RCPPS
1  foreach $itemset\ Px \in ExtensionsOfP$ do
2      $numSeq(Px) \leftarrow |\{s|sup(Px,s) \leq maxSup \wedge stanDev(Px,s) \leq$
         $maxStd \wedge bond(Px,s) \geq minBond \wedge s \in D\}|$;
3      $ra(Px) \leftarrow numSeq(Px)/|D|$;
4      if $ra(Px) \neq 0 \wedge ra(Px) \geq minRa$ then output $Px$;
5      $ExtensionsOfPx \leftarrow \emptyset$;
6      foreach $itemset\ Py \in ExtensionsOfP$ such that $y \succ x$ do
7          $Pxy \leftarrow Px \cup Py$;
8          $Pxy.RCPPS\text{-}list \leftarrow$ Construct($Px.RCPPS\text{-}list$, $Py.RCPPS\text{-}list$);
9          $numCand(Pxy) \leftarrow |\{s|bond(Pxy,s) \geq minBond \wedge s \in D\}|$;
10         $upBondRa(Pxy) \leftarrow numCand(Pxy)/|D|$;
11         if $upBondRa(Pxy) \neq 0 \wedge upBondRa(Pxy) \geq minRa$ then
12             $ExtensionsOfPx \leftarrow ExtensionsOfPx \cup Pxy$;
13         end
14     end
15 end
16 Find ($Px$, $ExtensionsOfPx$, $D$, $maxSup$, $maxStd$, $minBond$, $minRa$);
```

To illustrate how the MRCPPS algorithm is applied, a detailled example is next presented for mining the RCPP in the sequence database of Table 3. Consider that the parameters are set as $minSup = 2$, $maxStd = 1$, $minBond = 0.6$ and $minRa = 0.6$. The following steps are performed.

1. The algorithm reads the database and creates the RCPPS-lists of each item. For example, the RCPPS-lists of itemsets $\{b\}$ and $\{e\}$ are shown in Fig. 3.
2. The *upBondRa* value of each item is calculated. For the item $\{b\}$, it is found that $upBondRa(b) = 1$, and thus its extensions should be considered. The items $\{a\}$, $\{c\}$, $\{d\}$ and $\{e\}$ also satisfy this condition. All these items are thus put in the variable $I^*$ and the $Find$ procedure is called with them.
3. The $Find$ procedure first checks item $b$. Because $ra(\{b\}) = 0.25 \geq minRa$, the itemset $\{b\}$ is output as a RCPP.
4. Then, extensions of $\{b\}$ are considered such as $\{b, e\}$. The RCPPS-list of itemset $\{b, e\}$ is built by applying the *Construct* procedure with the RCPPS-lists of $\{b\}$ and $\{e\}$. During this process, the list of sequences where both items are periodic and have an *upBondRa* value no less than *minRa* are identified. The obtained RCPPS-list of $\{b, e\}$ is shown in Fig. 3. Because, $upBondRa(\{b, e\}) = 0.75$, the *Find* method is invoked to next search for extensions of $\{b, e\}$.
5. The *Find* procedure finds that $ra(\{b, e\}) = 0.75 \geq minRa$ and thus the itemset $\{b, e\}$ is output as a RCPP. The search then continues in a similar way to process other itemsets until all RCPP have been found.

## 5 Research Opportunities

There are several research opportunities related to discovering periodic patterns in multiple sequences. A reason is that researchers working on pattern mining have up to now mostly focused on finding periodic patterns in a single sequence. A first research opportunity is to design more efficient algorithms in terms of runtime, memory usage and scalability. This can be done using novel data structures, search strategies, and optimizations, but also by developing paralllel versions of algorithms that can run on multi-thread, multi-core, GPU, or big data platforms. A second research opportunity is to adapt the definitions of periodic pattern to find other types of periodic patterns in multiple sequences. This can be done by using different pattern selection functions such as measures of stability [26, 30] but also by considering other types of patterns besides itemsets such as sequential patterns [14, 18, 24, 43, 61, 66, 67], episodes [6, 31], subgraphs [16, 17, 34, 36], trajectory patterns [70], and periodic patterns with gap constraints [64, 65]. A third research opportunity is to explore novel applications of periodic patterns. As discrete sequences are found in many domains, many applications can be considered such as smart homes [47], location prediction [63], sequence prediction [? ], clustering [10, 11, 44], and privacy-preserving data mining [52].

## 6 Conclusion

This chapter has presented an overview of two recent algorithms for discovering periodic patterns in a set of discrete sequences, named MPFPS and MRCPPS. The pseudocode of the algorithms were presented as well as detailed examples. Moreover, research opportunities were also discussed.

The original implementations and Java source code of the depth-first and breadth-first search version of MPFPS, and MRCPPS, as well as the datasets that have been used for evaluating these algorithms can be found at `http://www.philippe-fournier-viger.com/spmf/`, as part of the SPMF data mining library [23].

## References

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

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

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

[4] Amphawan, K., Lenca, P., Surarerks, A.: Mining top-$K$ periodic-frequent pattern from transactional databases without support threshold. In: Proc. Third International Conference on Advanced in Information Technology, pp. 18–29 (2009)

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

[6] Ao, X., Shi, H., Wang, J., Zuo, L., Li, H., He, Q.: Large-scale frequent episode mining from complex event sequences with hierarchies. ACM Transactions on Intelligent Systems and Technology (TIST) **10**(4), 1–26 (2019)

[7] Barsky, M., Kim, S., Weninger, T., Han, J.: Mining flipping correlations from large datasets with taxonomies. Proc. VLDB Endow. **5**, 370–381 (2011)

[8] Bogarín, A., Cerezo, R., Romero, C.: A survey on educational process mining. Wiley Interdisciplinary Reviews: Data Mining and Knowledge Discovery **8**(1), e1230 (2018)

[9] Chi, Y., Muntz, R.R., Nijssen, S., Kok, J.N.: Frequent subtree mining–an overview. Fundamenta Informaticae **66**(1-2), 161–198 (2005)

[10] Dinh, D.T., Huynh, V.N.: k-PbC: an improved cluster center initialization for categorical data clustering. Applied Intelligence pp. 1–23 (2020)

[11] Dinh, D.T., Huynh, V.N., Songsak, S.: Clustering mixed numerical and categorical data with missing values. Information Sciences (2021)

[12] Dinh, D.T., Le, B., Fournier-Viger, P., Huynh, V.N.: An efficient algorithm for mining periodic high-utility sequential patterns. Applied Intelligence **48**(12), 4694–4714 (2018)

[13] Dinh, T., Huynh, V.N., Le, B.: Mining periodic high utility sequential patterns. In: Proc. 2017 Internal Conference on Intelligent Information and Database Systems, pp. 545–555. Springer (2017)

[14] Dinh, T., Quang, M.N., Le, B.: A novel approach for hiding high utility sequential patterns. In: Proceedings of the 6th International Symposium on Information and Communication Technology, pp. 121–128 (2015)

[15] Feng, Z., Zhu, Y.: A survey on trajectory data mining: Techniques and applications. IEEE Access **4**, 2056–2067 (2016)

[16] Fournier-Viger, P., Cheng, C., Lin, J.C.W., Yun, U., Kiran, R.U.: Tkg: Efficient mining of top-k frequent subgraphs. In: Proc. 7th International Conference on Big Data Analytics, pp. 209–226. Springer (2019)

[17] Fournier-Viger, P., He, G., Cheng, C., Li, J., Zhou, M., Lin, J.C.W., Yun, U.: A survey of pattern mining in dynamic graphs. Wiley Interdisciplinary Reviews: Data Mining and Knowledge Discovery p. e1372 (2020)

[18] Fournier-Viger, P., Li, J., Lin, J.C.W., Truong, T.: Discovering low-cost high utility patterns. Data Science and Pattern Recognition **4**(2), 50–64 (2020)

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

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

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

[22] Fournier-Viger, P., Lin, J.C.W., Duong, Q.H., Dam, T.L.: Phm: mining periodic high-utility itemsets. In: Proc. 16th Industrial conference on data mining, pp. 64–79. Springer (2016)

[23] Fournier-Viger, P., Lin, J.C.W., Gomariz, A., Gueniche, T., Soltani, A., Deng, Z., Lam, H.T.: The spmf open-source data mining library version 2. In: Joint European conference on machine learning and knowledge discovery in databases, pp. 36–40. Springer (2016)

[24] Fournier-Viger, P., Lin, J.C.W., Kiran, U.R., Koh, Y.S.: A survey of sequential pattern mining. Data Science and Pattern Recognition **1**(1), 54–77 (2017)

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

[26] Fournier-Viger, P., Wang, Y., Yang, P., Lin, J.C.W., Unil, Y.: A survey of sequential pattern mining. Applied Intelligence (2021)

[27] Fournier-Viger, P., Yang, P., Kiran, R.U., Ventura, S., Luna, J.M.: Mining local periodic patterns in a discrete sequence. Information Sciences **544**, 519–548 (2021)

[28] Fournier-Viger, P., Yang, P., Li, Z., Lin, J.C.W., Kiran, R.U.: Discovering rare correlated periodic patterns in multiple sequences. Data Knowl. Eng. (126), 101,733 (2020). DOI 10.1016/j.datak.2019.101733

[29] Fournier-Viger, P., Yang, P., Lin, J.C.W., Duong, Q.H., Dam, T., Sevcik, L., Uhrin, D., Voznak, M.: Discovering periodic itemsets using novel periodicity measures. Advances in Electrical and Electronic Engineering **17**(1), 33–44 (2019). DOI 10.15598/aeee.v17i1.3185

[30] Fournier-Viger, P., Yang, P., Lin, J.C.W., Kiran, R.U.: Discovering stable periodic-frequent patterns in transactional data. In: Proc. 32nd Intern. Conf. on Industrial, Engineering and Other Applications of Applied Intelligent Systems, pp. 230–244. Springer (2019)

[31] Fournier-Viger, P., Yang, Y., Yang, P., Lin, J.C.W., Yun, U.: Tke: Mining top-k frequent episodes. In: Proc. 33rd Intern. Conf. on Industrial, Engineering and Other Applications of Applied Intelligent Systems. Springer (2020)

[32] Fournier-Viger, P., Zhang, Y., Lin, J.C.W., Dinh, D.T., Bac Le, H.: Mining correlated high-utility itemsets using various measures. Logic Journal of the IGPL **28**(1), 19–32 (2020)

[33] Geng, L., Hamilton, H.J.: Interestingness measures for data mining: A survey. ACM Comput. Surv. **38**, 9 (2006)

[34] Halder, S., Samiullah, M., Lee, Y.K.: Supergraph based periodic pattern mining in dynamic social networks. Expert Systems with Applications **72**, 430–442 (2017)

[35] Henzgen, S., Hüllermeier, E.: Mining rank data. In: Proc. 17th International Conference on Discovery Science, pp. 123–134. Springer (2014)

[36] Jiang, C., Coenen, F., Zito, M.: A survey of frequent subgraph mining algorithms. Knowledge Engineering Review **28**, 75–105 (2013)

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

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

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

[40] Koh, Y.S., Ravana, S.D.: Unsupervised rare pattern mining: a survey. ACM Transactions on Knowledge Discovery from Data (TKDD) **10**(4), 1–29 (2016)

[41] Koh, Y.S., Rountree, N.: Finding sporadic rules using apriori-inverse. In: Proc. 9th Pacific-Asia Conference, PAKDD 2005, pp. 97–106. Springer (2005)

[42] Kumar, V., Kumari, V.: Incremental mining for regular frequent patterns in vertical format. International Journal of Engineering and Technology **5**(2), 1506–1511 (2013)

[43] Le, B., Huynh, U., Dinh, D.T.: A pure array structure and parallel strategy for high-utility sequential pattern mining. Expert Systems with Applications **104**, 107–120 (2018)

[44] Lu, N.V., Vuong, T.N., Dinh, D.T.: Combining correlation-based feature and machine learning for sensory evaluation of saigon beer. International Journal of Knowledge and Systems Science (IJKSS) **11**(2), 71–85 (2020)

[45] Lu, Y., Richter, F., Seidl, T.: Efficient infrequent pattern mining using negative itemset tree. In: Complex Pattern Mining, pp. 1–16. Springer (2020)

[46] Luna, J.M., Fournier-Viger, P., Ventura, S.: Frequent itemset mining: A 25 years review. Wiley Interdisciplinary Reviews: Data Mining and Knowledge Discovery **9**(6), e1329 (2019)

[47] Mukhlash, I., Yuanda, D., Iqbal, M.: Mining fuzzy time interval periodic patterns in smart home data. International Journal of Electrical and Computer Engineering **8**(5), 3374 (2018)

[48] Nawaz, S., Fournier-Viger, P., Shojaee, A., Fujita, H.: Using artificial intelligence techniques for covid-19 genome analysis. Applied Intelligence (2021)

[49] Nofong, V.M.: Discovering productive periodic frequent patterns in transactional databases. Annals of Data Science **3**(3), 235–249 (2016)

[50] Nofong, V.M.: Fast and memory efficient mining of periodic frequent patterns. In: Proceedings of the 10th Asian Conference on Intelligent Information and Database Systems, pp. 223–232. Springer (2018)

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

[52] Quang, M.N., Dinh, T., Huynh, U., Le, B.: MHHUSP: An integrated algorithm for mining and Hiding High Utility Sequential Patterns. In: Proceedings of the 8th International Conference on Knowledge and Systems Engineering, pp. 13–18 (2016)

[53] Rashid, M.M., Gondal, I., Kamruzzaman, J.: Regularly frequent patterns mining from sensor data stream. In: Proc. 20th International Conference on Neural Information Processing, pp. 417–424. Springer (2013)

[54] Rashid, M.M., Karim, M.R., Jeong, B.S., Choi, H.J.: Efficient mining regularly frequent patterns in transactional databases. In: Proc. 17th International Conference on Database Systems for Advanced Applications, pp. 258–271. Springer (2012)

[55] Shekhar, S., Evans, M.R., Kang, J.M., Mohan, P.: Identifying patterns in spatial information: A survey of methods. Wiley Interdisciplinary Reviews: Data Mining and Knowledge Discovery **1**(3), 193–214 (2011)

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

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

[58] Szathmary, L., Napoli, A., Valtchev, P.: Towards rare itemset mining. In: 19th IEEE International Conference on Tools with Artificial Intelligence, vol. 1, pp. 305–312. IEEE (2007)

[59] Szathmary, L., Valtchev, P., Napoli, A., Godin, R.: Efficient vertical mining of minimal rare itemsets. In: Proc. 9th Intern. Conf. Concept Lattices and Their Applications, pp. 269–280. Citeseer (2012)

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

[61] Truong, T., Tran, A., Duong, H., Le, B., Fournier-Viger, P.: Ehusm: Mining high utility sequences with a pessimistic utility model. Data Science and Pattern Recognition **4**(2), 65–83 (2020)

[62] Venkatesh, J.N., Kiran, R.U., Reddy, P.K., Kitsuregawa, M.: 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, pp. 55–70 (2016). DOI 10.1007/978-3-319-44403-1_4

[63] Wong, M.H., Tseng, V.S., Tseng, J.C., Liu, S.W., Tsai, C.H.: Long-term user location prediction using deep learning and periodic pattern mining. In: Proc. 12th Conference on Advanced Data Mining and Applications, pp. 582–594. Springer (2017)

[64] Wu, Y., Shen, C., Jiang, H., Wu, X.: Strict pattern matching under non-overlapping condition. Science China Information Sciences **60**(1), 1–16 (2017)

[65] Wu, Y., Tong, Y., Zhu, X., Wu, X.: Nosep: Nonoverlapping sequence pattern mining with gap constraints. IEEE transactions on cybernetics **48**(10), 2809–2822 (2017)

[66] Wu, Y., Wang, L., Ren, J., Ding, W., Wu, X.: Mining sequential patterns with periodic wildcard gaps. Applied intelligence **41**(1), 99–116 (2014)

[67] Wu, Y., Zhu, C., Li, Y., Guo, L., Wu, X.: Netncsp: Nonoverlapping closed sequential pattern mining. Knowledge-based systems **196**, 105,812 (2020)

[68] Younes, N.B., Hamrouni, T., Yahia, S.B.: 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, pp. 189–204 (2010). DOI 10.1007/978-3-642-16184-1_14

[69] Zaki, M.J.: Efficiently mining frequent trees in a forest: Algorithms and applications. IEEE transactions on knowledge and data engineering **17**(8), 1021–1035 (2005)

[70] Zhang, D., Lee, K., Lee, I.: Hierarchical trajectory clustering for spatio-temporal periodic pattern mining. Expert Systems with Applications **92**, 1–11 (2018)