Finding Periodic Patterns in Big Data

R. Uday Kiran[1] and Masaru Kitsuregawa[1,2]

[1]University of Tokyo, Tokyo, Japan
[2]National Institute of Informatics, Tokyo, Japan
{uday_rage,kitsure}@tkl.iis.u-tokyo.ac.jp

Abstract. Periodic pattern mining is an important model in data mining. It typically involves discovering all patterns that are exhibiting either complete or partial cyclic repetitions in a dataset. The problem of finding these patterns has been widely studied in time series and (temporally ordered) transactional databases. This paper contains these studies along with their advantages and disadvantages. This paper also discusses the usefulness of periodic patterns with two real-world case studies. The first case study describes the useful information discovered by periodic patterns in an aviation dataset. The second case study describes the useful information discovered by periodic patterns pertaining to users' browsing behavior in an eCommerce site.

The tutorial will start by describing the frequent pattern model and the importance of enhancing this model with respect to time dimension. Next, we discuss the basic model of finding periodic patterns in time series, describe its limitations, and the approaches suggested to address these limitations. We next discuss the basic model of finding periodic patterns in a transactional database, describe its limitations, and the approaches suggested to address them. Finally, we end this tutorial with the real-world case studies that demonstrate the usefulness of these patterns.

Keywords: Data mining, knowledge discovery in databases, frequent patterns and periodic patterns

1 Introduction

Time and frequency are two most important dimensions to determine the interestingness of a pattern in a given data set. Periodic patterns are an important class of regularities that exist in a data set with respect to these two dimensions. Periodic pattern mining involves discovering all patterns that are exhibiting either complete or partial cyclic repetitions in a data set [1, 2]. Finding these patterns is a significant task with many real-world applications. Examples include finding co-occurring genes in biological data sets [3], improving the performance of recommender systems [4], intrusion detection in computer networks [5], and finding events in Twitter [6]. A classic application to illustrate the usefulness of these patterns is market-basket analysis. It analyzes how regularly the sets of items are being purchased by the customers. An example of a periodic pattern is as follows:

$$\{Bed,\ Pillow\} \quad [support = 10\%,\ \ period = 1\ hour].$$

The above pattern says that 10% of customers have purchased the items '*Bed*' and '*Pillow*' at least once in every hour.

The problem of finding periodic patterns has been studied in [1–3, 7–16]. Some of these approaches consider input data as time series [2, 3, 7–11], while others consider data as an enhanced transactional database having time attribute [1, 12–16]. In this paper, we study all of these approaches with respect to the following topics:

1. **Data representation.** How a periodic pattern model considers input data? What are the implicit assumptions pertaining to the frequency and periodic behavior of the items within the data?
2. **Computational expensiveness.** What is the size of search space? What properties are used to reduce the search space efficiently?
3. **Mining rarity.** In many real-world databases, some items appear very frequently in the data, while others appear rarely. The knowledge pertaining to rare items is often of great interest and high value. However, finding this knowledge is challenging due to infrequent appearances of rare items. This problem is known as the *rare item problem.* We discuss how some of the periodic pattern models are trying to address this problem.

The rest of the paper is organized as follows. Section 2 describes the approaches for finding periodic patterns in time series. Section 3 describes the approaches for finding periodically occurring frequent patterns in a transactional database. Sections 4 reports on the experimental results. Finally, Section 5 concludes the paper.

2 Periodic pattern mining in time series

In this section, we first describe the basic model of periodic patterns. We next discuss the limitations of this model. Finally, we describe the approaches that try to address these limitations.

2.1 The basic model of periodic patterns

Time series is a collection of events obtained from sequential measurements over time. Han et al. [2] have studied the periodic behavior of patterns in a series, and discussed a model to find periodic patterns. The model is as follows:

Let I be the set of items and D be a set of transactions (or a database), where each transaction t is a set of items such that $t \subseteq I$. The time series S represents the gathering of n timestamped databases into a single database, i.e., $S = D_1, D_2, \cdots, D_n$, $1 \leq n$. Let the symbol * denote a wild (or do not care) character, which can match any single set of items. The pattern $s = s_1, \cdots, s_p$ is a non-empty sequence over $(2^I - \{\emptyset\}) \cup \{*\}$. Let $|s|$ denote the length (or *period*) of the pattern s. Let the I-length of $s = s_1 \cdots s_p$ be the number of s_i which contains letters from I. A pattern with I-length k is also called a k-pattern. Moreover, a subpattern of a pattern $s = s_1 \cdots s_p$ is a pattern $s' = s'_1 \cdots s'_p$ such that s and s' have the same length, and $s'_i \subseteq s_i$for every position i where $s'_i \neq *$. The *support* of a pattern s in S is denoted as $Sup(x) = |\{i|0 \leq i < m,\ and\ the\ string\ s\ is\ true\ in\ D_{i|s|+1} \cdots D_{i|s|+|s|}\}|$, where m is the maximum number of *periods* of length s. Each segments of the form $D_{i|s|+1} \cdots D_{i|s|+|s|}$, where $0 \leq i < m$, is called a period segment. The pattern s is said to be a periodic pattern if $Sup(s) \geq minSup$, where $minSup$ represents the user-defined minimum support threshold value.

Example 1. Given the time series $S = a\{bc\}baebace$, $I = \{a, b, c, e\}$. If the user-defined *period* is 3, S is divided into three periodic-segments: $D_1 = a\{bc\}b$, $D_2 = aeb$ and $D_3 = ace$. Let $a * b$ be a pattern. The length of this pattern is 3, and its I-length is 2 (i.e., it contains only two items within this pattern). Therefore, we represent this pattern as 2-pattern. This pattern appears in the periodic-segments of D_1 and D_2. Therefore, its *support* count is 2. If the user-defined $minSup$ is 2, then $a \star b$ represents a periodic pattern.

2.2 The limitations of basic model

Aref et al. [17] have extended the Han's model to incremental mining of periodic patterns. Yang et al. [11] have studied the change in periodic behavior of a pattern due to the influence of noise, and enhanced the basic model to discover a class of periodic patterns known as asynchronous periodic patterns. Zhang et al. [3] have enhanced the basic model to discover periodic patterns in character sequences like protein data. The popular adoption and successful industrial application of this basic periodic pattern model suffers from the following obstacles:

1. **Computationally expensive model:**
   - In the basic periodic pattern model, a pattern represents sets of items. Therefore, the search space of this model is $\sum_{i=1}^{p} n^p$, where n and p represent the total number of items in a series and the user-defined *period*, respectively. This search space is typically much higher than the frequent pattern model that has the search space of $2^n - 1$.
   - Periodic patterns satisfy the *anti-monotonic property* [18]. That is, all non-empty subsets of a periodic pattern are also periodic patterns. However, this property is insufficient to make the periodic pattern mining practical or computationally inexpensive in real-life. The reason is that number of frequent i-patterns shrink slowly (when $i > 1$) as i increases in time series data. The slow speed of decrease in the number of frequent i-patterns is due to a strong correlation between frequencies of patterns and their sub-patterns [19].

  - Overall, the huge search space followed by the inability to reduce the search space using *anti-monotonic property* makes the model computationally expensive or impractical in real-world applications.

2. **Sparsity problem:** The basic model of periodic patterns uses the wild character '$\star$' to represent an event within a pattern. This leads to the sparsity problem, which involves many discovered patterns having a large number of wild characters with very few events. For example, $a \star\star\star\star\star\star\star\star\star\star\star\star\star bc \star\star\star\star\star\star\star\star a \star\star\star\star\star\star\star\star\star b$. This problem makes the discovered patterns impracticable in applications.
3. **The rare item problem:** Since only a single *period* and *minSup* are used for the whole data, the model implicitly assumes that all items in the data have same periodic behavior and uniform frequency. However, this is seldom not the case in many real-world data sets. In many data sets, some items appear very frequently in the data, while others rarely appear. Moreover, rare items typically occur with long *periods* (i.e., inter-arrival times) as compared against the frequent items. Henceforth, finding periodic patterns with a single *period* and *minSup* leads to the following two problems:
   - If the *period* is set too short and/or the *minSup* is set too high, we will miss the periodic patterns involving rare items.
   - In order to find the periodic patterns involving both frequent and rare items, we have to set a long *period* and a low *minSup*. However, this may result in combinatorial explosion, producing too many patterns, because frequent items can combine with one another in all possible ways and many of them may be meaningless.
4. **Methodology to specify period:** An open problem of this model is the methodology to specify *period* that can capture the heterogeneous temporal behavior of all items in a series effectively.
5. **Inability to consider temporal information of events:** Han's model considers time series as a symbolic sequence. As a result, this model does not take into account the actual temporal information of events within a sequence [5].

2.3 Research efforts to address the limitations

Han et al. [19] have introduced "maximum sub-pattern hit set property" to reduce the computational cost of finding these patterns . Based on this property, a tree-based algorithm, called max-subpattern tree, has been proposed to discovered periodic patterns effectively. A max-subpattern tree takes the candidate max-pattern, C_{max}, as the root node. Each subpattern of C_{max} with one non-* letter missing is a direct child node of the root. The tree expands recursively according to the following rules. A node w may have a set of children if it contains more 2 non-* letters. Then the tree from the root of the tree is constructed and the missing non-* letters are checked in order to find the corresponding node. The count increases by 1 if the node w is found. Otherwise, a new node w (with count 1) and its missing ancestor nodes (only those on the path to w, with count

0) are created. If one exists, it (or them) is (or are) inserted into the corresponding place(s) of the tree. After a max-subpattern tree has been built, the tree is scanned to find frequency counts of the candidate patterns and eliminate the non-frequent ones. Notice that the frequency count of a node is the sum of the count of itself and those of all of the reachable ancestors. If the derived frequent pattern set is empty, then return. For more details, refer to the study by [19].

Zhang et al. [3] have tried to address the sparsity problem by limiting the gap (i.e., number of wild characters allowed between two itemsets) within a pattern. Yang et al. [7] have used "*information gain*" to address the rare item problem in periodic pattern mining. The discovered patterns are known as *surprising patterns*. Alternatively, Chen et al. [8] have tried to address this problem by finding periodic patterns using multiple *minSups* [20]. In this approach, each item in the series is specified with a *support* constraint known as *minimum item support* (MIS). The *minSup* for a pattern is represented with the lowest MIS value of its items. That is,

$$minSup(s) = minmum(MIS(i_j)|\forall i_j \in s)$$

where, $MIS(i_j)$ represents the user-specified minimum item support for an item $i_j \in s$. The usage of *minimum item support* enables us to achieve the goal of having higher *minSup* for patterns that only involve frequent items, and having lower *minSup* for patterns that involve rare items.

Berberidis et al. [9] have discussed a methodology to specify approximate *period* using Fast Fourier Transformations and auto-correlation. Unfortunately, this method may miss some frequent periods and it requires a separate pass to scan the data sequence to compute the periods. Moreover, for each item, it needs to compute, at a high cost, the circular auto-correlation value for different periods in order to determine whether the period is frequent or not. Cao et al. [10] have proposed a computationally inexpensive approach to determine *period* with an implicit assumption that periodic pattern can be approximately expressed by an arithmetic series together with a support indicate about its frequency.

All of the above mentioned approaches consider time series as a symbolic sequence, and therefore, do not take into account the actual temporal information of the items in the data. In the next section, we discuss the approaches that take into account the temporal information of the items in the data.

3 Periodic pattern mining in transactional databases

In this section, we first describe the basic model of periodic-frequent patterns. We next discuss the limitations of this model, and describe the efforts made in the literature to address these limitations.

3.1 The basic model of periodic-frequent patterns

Ozden et al. [1] have enhanced the transactional database by a time attribute that describes the time when a transaction has appeared, investigated the periodic behavior of the patterns to discover cyclic association rules. In this study, a database is fragmented into non-overlapping subsets with respect to time. The association rules that are appearing in at least a certain number of subsets are discovered as cyclic association rules. By fragmenting the data and counting the number of subsets in which a pattern occurs greatly simplifies the design of the mining algorithm. However, the drawback is that patterns (or association rules) that span multiple windows cannot be discovered.

Tanbeer et al. [12] have discussed a simplified model to discover periodically occurring frequent patterns, i.e., periodic-frequent patterns, in a transactional database. The model is as follows:

Let I be a set of items. Let $X \subseteq I$ be a pattern (or an itemset). A pattern containing k number of items is known as a k-pattern. A **transaction**, $tr = (ts, Y)$, is a tuple, where $ts \in R$ represents the timestamp and Y is a pattern. A **transactional database** TDB over I is a set of transactions, $TDB = \{tr_1, \cdots, tr_m\}$, $m = |TDB|$, where $|TDB|$ is the size of the TDB in total number of transactions. For a transaction $tr = (ts, Y)$, such that $X \subseteq Y$, it is said that X occurs in tr and such a timestamp is denoted as ts^X. Let $TS^X = \{ts_k^X, \cdots, ts_l^X\}$, where $1 \leq k \leq l \leq m$, denote an **ordered set of timestamps** at which X has occurred in TDB.

Example 2. Table 1 shows the transactional database with each transaction uniquely identifiable with a timestamp (ts). All transactions in this database have been ordered with respect to their timestamps. This database do not contain any transaction with timestamps 6 and 9. However, it has to be noted that these two timestamps still contribute in determining the periodic interestingness of a pattern. The set of all items in this database, $I = \{a, b, c, d, e, f, g, h\}$. The set of items '$a$' and '$b$', i.e., '$ab$' is a pattern. This pattern contains only two items. Therefore, this is a 2-pattern. This pattern appears at the timestamps of $1, 2, 5, 7$ and 10. Therefore, $TS^{ab} = \{1, 2, 5, 7, 10\}$.

The above measures, *support* and *all-confidence*, determine the interestingness of a pattern in frequency dimension. We now describe the measures to determine the interestingness of a pattern in time dimension.

Table 1. A transactional database

ts	Items	ts	Items	ts	Items	ts	Items	ts	Items
1	ab	3	$cdgh$	5	ab	8	cd	11	cdg
2	abd	4	cef	7	$abce$	10	$abdef$	12	aef

Definition 1. ***(Support of a pattern*** X***.)*** *The number of transactions containing* X *in* TDB *(i.e., the size of* TS^X*) is defined as the support of* X *and denoted as sup*(X)*. That is, sup*$(X) = |TS^X|$*.*

Example 3. The support of 'ab' in Table 1 is the size of TS^{ab}. Therefore, $sup(ab) = |TS^{ab}| = |1, 2, 5, 7, 10| = 5$.

Definition 2. ***(A frequent pattern** X.)* *The pattern X is said to be frequent if its support satisfies the user-specified **minimum support** (minSup) constraint. That is, if $SUP(X) \geq minSup$, then X is a frequent pattern.*

Example 4. Continuing with the previous example, if the user-specified $minSup = 3$, then ab is a frequent pattern as $sup(ab) \geq minSup$.

Definition 3. ***(A period of** X)* *Given $TS^X = \{ts_a^X, ts_b^X, \cdots, ts_c^X\}$, $1 \leq a \leq b \leq c \leq |TDB|$, a period of X, denoted as p_k^X, is calculated as follows:*

- $p_1^X = ts_a^X - ts_{ini}$, *where $ts_{ini} = 0$ denotes the initial timestamp of all transactions in TDB.*
- $p_k^X = ts_q^X - ts_p^X$, $1 < k < Sup(X) + 1$, *where ts_p^X and ts_q^X, $a \leq p < q \leq c$, denote any two consecutive timestamps in TS^X.*
- $p_{sup(X)+1}^X = ts_{fin} - ts_c^X$, *where ts_{fin} denotes the final timestamp of all transactions in TDB.*

Example 5. In Table 1, the pattern ab has initially appeared at the timestamp of 1. Therefore, the initial period of ab, i.e., $p_1^{ab} = 1$ $(= 1 - ts_{ini})$. Similarly, other periods of this pattern are: $p_2^{ab} = 1$ $(= 2-1)$, $p_3^{ab} = 3$ $(= 5-2)$, $p_4^{ab} = 2$ $(= 7-5)$, $p_5^{ab} = 3$ $(= 10-7)$ and $p_6^{ab} = 2$ $(= ts_{fin} - 10)$.

The terms 'ts_{ini}' and 'ts_{fin}' play a key role in determining the periodic appearance of X in the entire database. Let $P^X = \{p_1^X, p_2^X, \cdots, p_k^X\}$, $k = Sup(X) + 1$, denote the set of all periods of X in TDB. The first *period* in P^X (i.e., p_1^X) provides useful information pertaining to time taken for initial appearance of X in TDB. The last *period* in P^X (i.e., p_k^X) provides useful information pertaining to time elapsed after the final appearance of X in TDB. Other periods in P^X provide information pertaining to inter-arrival times of X in TDB.

The occurrence intervals, defined as above, gives information of appearance behavior of a pattern. The largest occurrence period of a pattern provides the upper limit of its periodic occurrence characteristic. Hence, the measure of the characteristic of a pattern of being periodic in a TDB can be defined as follows.

Definition 4. ***(Periodicity of** X.)* *Let P^X denote the set of all periods of X in TDB. The maximum period in P^X is defined as the **periodicity** of X and is denoted as $per(X)$, i.e., $per(X) = max(p_i^X | \forall p_i^X \in P^X)$.*

Example 6. Continuing with the previous example, $P^{ab} = \{1, 1, 3, 2, 3, 2\}$. Therefore, the *periodicity* of ab, i.e., $per(ab) = max(1, 1, 3, 2, 3, 2) = 3$.

Definition 5. ***(A periodic-frequent pattern** X.)* *The pattern X is said to be periodic-frequent if $sup(X) \geq minSup$ and $per(x) \leq maxPer$.*

Example 7. Continuing with the previous example, if the user-specified $maxPer = 3$, then ab is a periodic-frequent pattern as $sup(ab) \geq minSup$ and $per(ab) \leq maxPer$.

3.2 The performance issues of the periodic-frequent pattern model

Unlike the periodic pattern mining models in time series, periodic-frequent pattern mining model takes into account the temporal information of the items within the data. Moreover, the latter model do not suffer from the *sparsity problem*. This model is also computationally inexpensive than the periodic pattern model, because its search space of $2^n - 1$, where n is the total number of items within the database, is much less than search space of periodic pattern model. However, this model still suffers from the following performance issues:

1. **The rare item problem.** The usage of single $minSup$ and $maxPer$ leads to the rare item problem.
2. **Inability to discover partial periodic-frequent patterns.** Since $maxPrd$ controls the maximum inter-arrival time of a pattern in the entire database, this model discovers only full periodic-frequent patterns (i.e., only those frequent patterns that have exhibited complete cyclic repetitions in a database). As the real-world is imperfect, partial periodic patterns are ubiquitous in real-world databases. This model fails to discover these patterns.

3.3 Research efforts to address the performance issues

To address the rare item problem, Uday et al. [13] have proposed an enhanced model to discover periodic-frequent patterns using multiple $minSups$ and $maxPrds$. In this model, the $minSup$ and $maxPrd$ for a pattern are represented as follows:

$$\begin{aligned} minSup(X) &= min(minIS(i_j) | \forall i_j \in X) \\ & and \\ maxPer(X) &= max(maxIP(i_j) | \forall i_j \in X). \end{aligned} \tag{1}$$

Where $minIS(i_j)$ and $maxIP(i_j)$ respectively represent the user-specified *minimum item support* and *maximum item periodic* for an item $i_j \in X$. This model facilitates the user to specify a low $minSup$ and a high $maxPrd$ for a pattern containing rare items, and high $minSup$ and a low $maxPrd$ for a pattern containing only frequent items.

The periodic-frequent patterns discovered by [13] do not satisfy the anti-monotonic property. This increases the search space, which in turn increases the computational cost of finding the periodic-frequent patterns. In other words, this enhanced model is impracticable in real-world very large databases. Akshat et al. [21] have proposed another model using the notion of multiple $minSups$ and $maxPrds$. In this model, the $minSup$ and $maxPer$ for a pattern are represented as follows:

$$\begin{aligned} minSup(X) &= max(minIS(i_j) | \forall i_j \in X) \\ & and \\ maxPer(X) &= min(maxIP(i_j) | \forall i_j \in X) \end{aligned} \tag{2}$$

The periodic-frequent patterns discovered by this model satisfy the anti-monotonic property. Therefore, this model is practicable in real-world databases.

Amphawan et al. [14] have used standard deviation of *periods* to determine the periodic interestingness of a pattern in the database. Uday et al. [22] have introduced a novel measure, called *periodic-ratio*, to discover partial periodic-frequent patterns in a database. The *periodic-ratio* of a pattern X is calculated as follows:

$$PR(x) = \frac{|IP^X|}{|P^X|}, \quad (3)$$

where $IP^X \subseteq P^X$, such that $\forall p_i^X \in IP^X, p_i^X \leq maxPeriod$. The term $maxPeriod$ refers to the user-defined maximum period threshold value.

4 Experimental Results

In this section, we discuss the usefulness of periodic-frequent patterns using two real-world (**Shop-4** and **Accidents**) databases. We use periodic-frequent pattern-growth++ (PF-growth++) [16] to discover periodic-frequent patterns. In this paper, we do not discuss the usefulness of periodic patterns discovered in time series data. The reasons are as follows: (i) there exists no publicly available real-world time series data and (ii) current periodic pattern mining algorithms do not consider temporal information of the items within a series.

4.1 Experimental setup

The PF-growth++ algorithm is written in GNU C++, and run on Ubuntu 14.04 machine having 16GB of RAM. The details of the databases are as follows:

1. **Shop-4 database.** A Czech company has provided clickstream data of seven online stores in the ECML/PKDD 2005 Discovery challenge [23]. For our experiment, we have considered the click stream data of product categories visited by the users in "Shop 4" (www.shop4.cz), and created a transactional database with each transaction representing the set of web pages visited by the people at a particular *minute interval.* The transactional database contains 59,240 transactions (i.e., 41 days of page visits) and 155 distinct items (or product categories).
2. **Accidents database.** The Federal Aviation Authority (FAA) has collected data pertaining to aircraft damages. In order to improve aviation safety, this data was made available in Aviation Safety Information Analysis and Sharing (ASIAS) system. The **Accidents** database is created from the data retrieved from ASIAS from 1-January-1978 to 31-December-2014 [24]. The raw data collected by FAA contains both numerical and categorical attributes. For our experiments, we have considered only categorical attributes, namely 'local event date,' 'event city,' 'event state,' 'event airport,' 'event type,' 'aircraft damage,' 'flight phase,' 'aircraft make,' 'aircraft model,' 'operator,' 'primary

flight type,' 'flight conduct code,' 'flight plan filed code' and 'PIC certificate type.' The missing values for these attributes are ignored while creating this database.

The statistical details of these two databases are provided in Table 2.

Table 2. Database statistics. The terms, T_{min}, T_{avg} and T_{max}, represent the minimum, average and maximum number of items within a transaction, respectively

Database	T_{min}	T_{avg}	T_{max}	Size	Items
Shop-4	1	2.4	82	59,240	155
Accidents	3	8.9	9	98,864	9,290

4.2 Generation of periodic-frequent patterns

Figure 1(a) and (b) shows the number of periodic-frequent patterns discovered in Shop-4 and Accidents database, respectively. The X-axis represents the $maxPer$ values used to discover periodic-frequent patterns. The Y-axis represents the number of periodic-frequent patterns generated at a particular $maxPer$ value. Each line in this figure represents the number of periodic-frequent patterns generated at a particular $minSup$ value. The $minSup$ used in our experiments are 0.01%, 0.06% and 0.11%. The reason for setting low $minSup$ values is to generate periodic-frequent patterns involving both frequent and rare items. The following observations can be drawn from these two figures:

1. The increase in $maxPrd$ results in the increase of periodic-frequent patterns. The reason is that increase in $maxPrd$ causes sporadically appearing patterns to be discovered as periodic-frequent patterns.
2. The increase in $minSup$ results in the decrease the periodic-frequent patterns, because it is difficult for the items to appear frequently with other items in the entire database.

4.3 Interesting patterns discovered in accidents database

We discuss the usefulness of periodic-frequent patterns discovered in Accidents database. Table 3 presents some of the interesting periodic-frequent patterns discovered in accidents database when $minSup = 0.11\%$ and $maxPer = 4\%$. The first pattern in this table conveys the useful information that 'substantial' damages to an aircraft have happened during 'general operating rules.' The frequency of this event is 1,899 and the maximum inter-arrival time of this event is 110 days (i.e., almost 4 months). The second pattern conveys the information that 'substantial' damages to an aircraft with 'private pilot' have happened during 'general operating rules.' The frequency of this event is 750, while the *periodicity* of this event is 219 days (i.e., around 7 months). The third pattern

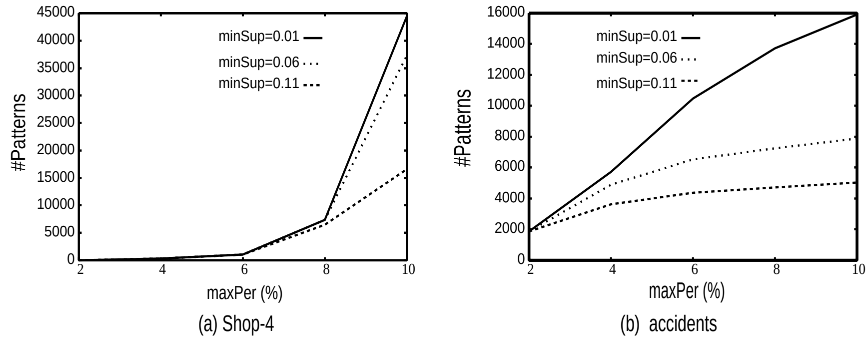

Fig. 1. Periodic-frequent patterns generated at different *minSup* and *maxPer* values

conveys the crucial information that within every 15 days, aircrafts of 'Cessna' airlines undergo a 'minor' accident. The fourth pattern provides the information that within every 22 days, aircrafts in 'California' airport witness a 'minor' accident. The fifth pattern provides the information that with an interval of 15 days, 16154 'minor' aircraft accidents pertaining to 'Cessna' airlines have happened during 'general operating rules'.

Table 3. Interesting patterns found in accidents database

S.No.	Patterns	Support	Periodicity
1	{General operating rules, Substantial}	1899	110
2	{General operating rules, private pilot, substantial}	750	219
3	{Cessna, Minor}	17,971	15
4	{California, Minor}	6,620	22
5	{General operating rules, Cessna, Minor}	16,154	15

4.4 Interesting patterns discovered in Shop-4 database

We now discuss the useful information discovered by periodic-frequent patterns in Shop-4 database. Table 4 presents some of the interesting patterns discovered in Shop-4 database. The *periodicity* of a pattern is expressed in *minutes*. These patterns were interesting, because they contain costly and durable goods, which are regularly viewed by the visitors.

5 Conclusions

This paper classifies current periodic pattern mining algorithms into two types: (*i*) finding periodic patterns in time series data and (*ii*) finding periodically occurring frequent patterns in temporally ordered transactional databases.This

Table 4. Interesting patterns discovered in Shop-4 database

S. No.	Patterns	*Support*	*Periodicity*
1	{{Built-in ovens, hobs, grills}, {Washer dryers}}	4861	1353
2	{{Built-in ovens, hobs, grills}, {Microwave ovens}}	2134	2112
3	{{Refrigerators, freezers, show cases}, {washer dryers}}	5628	1288
4	{Washing machines, washer dryers}	8342	1114

paper describes the basic model of periodic patterns in time series and transactional databases, and also discusses advantages and disadvantages of each model. Experimental results on real-world data sets demonstrate that periodic patterns can find useful information.

6 Acknowledgments

This work was supported by the Research and Development on Real World Big Data Integration and Analysis program of the Ministry of Education, Culture, Sports, Science, and Technology, JAPAN.

References

1. B. Özden, S. Ramaswamy, and A. Silberschatz, "Cyclic association rules," in *ICDE*, 1998, pp. 412–421.
2. J. Han, W. Gong, and Y. Yin, "Mining segment-wise periodic patterns in time-related databases." in *KDD*, 1998, pp. 214–218.
3. M. Zhang, B. Kao, D. W. Cheung, and K. Y. Yip, "Mining periodic patterns with gap requirement from sequences," *ACM Trans. Knowl. Discov. Data*, vol. 1, no. 2, Aug. 2007.
4. H. Stormer, "Improving e-commerce recommender systems by the identification of seasonal products," in *Twenty second Conference on Artificial Intelligence*, 2007, pp. 92–99.
5. S. Ma and J. Hellerstein, "Mining partially periodic event patterns with unknown periods," in *ICDE*, 2001, pp. 205–214.
6. R. U. Kiran, Shang, M. Toyoda, and M. Kitsuregawa, "Discovering recurring patterns in time series," in *EDBT*, 2015, p. To be appeared.
7. R. Yang, W. Wang, and P. Yu, "Infominer+: mining partial periodic patterns with gap penalties," in *ICDM*, 2002, pp. 725–728.
8. S.-S. Chen, T. C.-K. Huang, and Z.-M. Lin, "New and efficient knowledge discovery of partial periodic patterns with multiple minimum supports," *J. Syst. Softw.*, vol. 84, no. 10, pp. 1638–1651, Oct. 2011.
9. C. Berberidis, I. Vlahavas, W. Aref, M. Atallah, and A. Elmagarmid, "On the discovery of weak periodicities in large time series," in *PKDD*, 2002, pp. 51–61.
10. H. Cao, D. Cheung, and N. Mamoulis, "Discovering partial periodic patterns in discrete data sequences," in *Advances in Knowledge Discovery and Data Mining*, vol. 3056, 2004, pp. 653–658.
11. J. Yang, W. Wang, and P. S. Yu, "Mining asynchronous periodic patterns in time series data." *IEEE Trans. Knowl. Data Eng.*, pp. 613–628, 2003.

12. S. K. Tanbeer, C. F. Ahmed, B. S. Jeong, and Y. K. Lee, "Discovering periodic-frequent patterns in transactional databases," in *PAKDD*, 2009, pp. 242–253.
13. R. U. Kiran and P. K. Reddy, "Towards efficient mining of periodic-frequent patterns in transactional databases," in *DEXA (2)*, 2010, pp. 194–208.
14. K. Amphawan, P. Lenca, and A. Surarerks, "Mining top-k periodic-frequent pattern from transactional databases without support threshold," in *Advances in Information Technology*, 2009, pp. 18–29.
15. R. U. Kiran and P. K. Reddy, "An alternative interestingness measure for mining periodic-frequent patterns," in *DASFAA (1)*, 2011, pp. 183–192.
16. R. U. Kiran and M. Kitsuregawa, "Novel techniques to reduce search space in periodic-frequent pattern mining," in *DASFAA (2)*, 2014, pp. 377–391.
17. W. G. Aref, M. G. Elfeky, and A. K. Elmagarmid, "Incremental, online, and merge mining of partial periodic patterns in time-series databases," *IEEE TKDE*, vol. 16, no. 3, pp. 332–342, Mar. 2004.
18. R. Agrawal, T. Imieliński, and A. Swami, "Mining association rules between sets of items in large databases," in *SIGMOD*, 1993, pp. 207–216.
19. J. Han, G. Dong, and Y. Yin, "Efficient mining of partial periodic patterns in time series database," in *ICDE*, 1999, pp. 106–115.
20. B. Liu, W. Hsu, and Y. Ma, "Mining association rules with multiple minimum supports," in *KDD*, 1999, pp. 337–341.
21. A. Surana, R. U. Kiran, and P. K. Reddy, "An efficient approach to mine periodic-frequent patterns in transactional databases," in *PAKDD Workshops*, 2011, pp. 254–266.
22. R. U. Kiran and M. Kitsuregawa, "Discovering quasi-periodic-frequent patterns in transactional databases," in *BDA*, 2013, pp. 97–115.
23. "Weblog dataset," http://web.archive.org/web/20070713202946rn_1/lisp.vse.cz/challenge/CURRENT/.
24. "Faa accidents dataset," http://www.asias.faa.gov/pls/apex/f?p=100:1:0::NO.