







Article

Status Set Sequential Pattern Mining Considering Time Windows and Periodic Analysis of Patterns

Shenghan Zhou [†], Houxiang Liu , Bang Chen , Wenkui Hou [†], Xinpeng Ji , Yue Zhang, Wenbing Chang and Yiyong Xiao ^{*}

School of Reliability and Systems Engineering, Beihang University, Beijing 100191, China; zhoush@buaa.edu.cn (S.Z.); zy1914125@buaa.edu.cn (H.L.); bang@buaa.edu.cn (B.C.); houwk@buaa.edu.cn (W.H.); sy1914103@buaa.edu.cn (X.J.); Zhangyue1127@buaa.edu.cn (Y.Z.); changwenbing@buaa.edu.cn (W.C.)

^{*} Correspondence: xiaoyiyong@buaa.edu.cn; Tel.: +86-136-1119-7256

[†] These authors contributed equally to this work.

Abstract: The traditional sequential pattern mining method is carried out considering the whole time period and often ignores the sequential patterns that only occur in local time windows, as well as possible periodicity. Therefore, in order to overcome the limitations of traditional methods, this paper proposes status set sequential pattern mining with time windows (SSPMTW). In contrast to traditional methods, the item status is considered, and time windows, minimum confidence, minimum coverage, minimum factor set ratios and other constraints are added to mine more valuable rules in local time windows. The periodicity of these rules is also analyzed. According to the proposed method, this paper improves the Apriori algorithm, proposes the TW-Apriori algorithm, and explains the basic idea of the algorithm. Then, the feasibility, validity and efficiency of the proposed method and algorithm are verified by small-scale and large-scale examples. In a large-scale numerical example solution, the influence of various constraints on the mining results is analyzed. Finally, the solution results of SSPM and SSPMTW are compared and analyzed, and it is suggested that SSPMTW can excavate the laws existing in local time windows and analyze the periodicity of the laws, which solves the problem of SSPM ignoring the laws existing in local time windows and overcomes the limitations of traditional sequential pattern mining algorithms. In addition, the rules mined by SSPMTW reduce the entropy of the system.

Keywords: data mining; status set sequential pattern mining; time window; TW-Apriori algorithm; periodicity analysis



Citation: Zhou, S.; Liu, H.; Chen, B.; Hou, W.; Ji, X.; Zhang, Y.; Chang, W.; Xiao, Y. Status Set Sequential Pattern Mining Considering Time Windows and Periodic Analysis of Patterns. *Entropy* **2021**, *23*, 738. <https://doi.org/10.3390/e23060738>

Academic Editor: Boštjan Brumen

Received: 24 April 2021

Accepted: 8 June 2021

Published: 11 June 2021

Publisher's Note: MDPI stays neutral with regard to jurisdictional claims in published maps and institutional affiliations.



Copyright: © 2021 by the authors. Licensee MDPI, Basel, Switzerland. This article is an open access article distributed under the terms and conditions of the Creative Commons Attribution (CC BY) license (<https://creativecommons.org/licenses/by/4.0/>).

1. Introduction

Data mining is a research field that has gained increasing attention in recent years, and sequential pattern mining is an important subtopic of data mining research. However, the traditional sequential pattern mining method has the following limitations: the traditional method is to mine the database considering the whole time period, which may ignore the sequential patterns that only occur in the local time period. For example, in hot summers, heat stroke is a common disease, and severe heat stroke often produces hyponatremia, heat stroke and other complications. However, there is no causal relationship between the disease and its complications, but a correlation relationship; thus, the sequence pattern of “heatstroke → complications” is often difficult to uncover when looking at the whole study period, but it is easy to find these rules when summer or even a hot period is selected as the study time window. In addition, traditional sequential pattern mining only considers the ID of the item, which cannot find the relationship between items with a different status. Traditional methods only consider the support constraint, which may lead to the mining of sequential patterns with low regularity and practical value.

In order to solve the above problems, this paper proposes the status set sequential pattern mining method with time windows (SSPMTW). The constraints of the time window, minimum confidence, minimum coverage and minimum factor set ratio are introduced to mine the sequential patterns in the local time window, and the periodic analysis of these sequential patterns is carried out so that the mined rules are the focus of users and have more regularity and practical value. In addition, SSPMTW considers the status of the items so that sequential patterns with status attributes can be mined. Aiming at the method proposed in this paper, this paper improved the Apriori algorithm, and proposes a TW-Apriori algorithm so as to complete the status set sequential pattern mining with time window and analyze the sequential pattern periodically.

1.1. Related Work

1.1.1. Research on Sequence Pattern Mining Method

Scholars across the world have conducted in-depth research on sequence pattern mining methods. This paper will describe the research status of sequence pattern mining methods from the following four aspects.

Sequential Pattern Mining with Time Windows

Sequential pattern mining with time windows is rarely studied, but association rule mining with time windows has been studied by some researchers. Xiao et al. [1] studied association rule mining with time windows in real transaction databases and proposed a TW-Apriori algorithm for frequent itemset mining with time windows. Xiao et al. [2] proposed a variable neighborhood search algorithm (VNS) for mining frequent itemsets with maximum time windows. Xiao et al. [3] proposed the VNS algorithm to optimize the maximum frequent time window selection problem in the sequence database of association rules. Zabihi et al. [4] proposed a novel fuzzy sequential pattern algorithm with a sliding window constraint which permits elements of a pattern to span a set of transactions within a user-specified window. Therefore, loss of useful sequences is prevented in the search process.

Periodic Sequence Pattern Mining

Periodic association rules were first proposed by Ozden et al. [5], who studied the association rules with periodicity that appear repeatedly in the database. Li and Jitender [6] proposed a method of mining partial periodic association rules in a time series database, which can mine association rules over a periodic period of time without restricting them to a fixed rule and sequence. Manziba [7] proposed a periodic sequence pattern mining algorithm, which does not rely on the user to obtain periodic correlation values but can also mine all types of periodic patterns at the same time. Han et al. [8] has studied some periodic patterns in time series database and proposed relevant algorithms to mine them efficiently. Based on the temporal Apriori algorithm, Li and Ning et al. [9] mined association rules with year, month and day as a cycle. Yang et al. [10] believed that the interference of random noise may lead to the asynchrony of a periodic mode, so he put forward the asynchronous periodic mode model. Huang and Chang [11] studied mining asynchronous short period patterns in temporal database, and each time point contained multiple events. Lee and Jiang et al. [12] relaxed the periodicity to a fuzzy periodicity and proposed an algorithm to mine the fuzzy periodicity association rule.

Study on Constraints of Sequential Pattern Mining

The constraints of sequential pattern mining will affect the number and type of frequent itemsets and sequential patterns. How to set the constraints and thresholds reasonably is a key problem. In order to solve the problem that a constant support threshold may cause long itemsets and sequences to be ignored, Seno and Karypis [13] combined LPMiner and SLPMER algorithms to mine long frequent itemsets and frequent sequence patterns. When the length of frequent itemsets and frequent sequences is longer, the support

threshold is smaller. Wang and Karypis [14] proposed a BAMBOO algorithm to mine closed itemsets based on decreasing support. Yun and Leggett [15] proposed a WLPMiner algorithm to construct a constraint that decreases in support as the length increases.

Yun [16] proposed a weighted and interesting sequential pattern mining method based on similarity support or weight level. Yun and Leggett [17] proposed a weighted frequent pattern mining method whose support decreases with increasing sequence pattern length. Chang [18] proposed a weighted sequential pattern mining method that considers time intervals in sequential databases.

Masseglia et al. [19] raised the issue of incremental mining of sequence patterns. When new transactions or customers join the original database, the cost of frequent sequence pattern mining from an updated database is reduced by using previously mined useful information. Ng et al. [20] proposed a sequence pattern mining method that takes into account time intervals so as to predict the time interval for any two transactions of a frequent sequence.

Closed Sequence Pattern Mining

A closed sequence pattern refers to the absence of any other contained sequence pattern under the same support. It not only fully expresses the complete set of results but also has a more streamlined result without information decay. In order to solve the problem that the traditional closed sequential pattern mining algorithm will produce a large number of inefficient redundant patterns when the support threshold is low and the sequential patterns in the database are diverse, Jingsong Zhang et al. [21] proposed the CCSpan algorithm. Fabregue et al. [22] proposed the OrderSpan algorithm, which extracts a closed set of partially ordered patterns from a sequence database. Moreover, the OrderSpan is extended by using efficient optimization methods used in the field of sequential pattern mining.

1.1.2. Research on Sequence Pattern Mining Algorithm

R. Agrawal and R. Srikant [23] proposed an algorithm based on the Apriori feature, which considers that all non-empty subsequences of the sequence pattern are sequence patterns. Based on this property, some Apriori-like algorithms have been proposed successively, which mainly contain two mining ideas: (1) horizontal data format mining ideas: the AprioriAll, AprioriSome, and DynamicSome algorithms proposed by R. Srikant and R. Agrawal [24] and the GSP algorithm proposed by F. Masseglia et al. [25]; (2) vertical data format mining ideas, such as the classical SPADE algorithm proposed by M. Zaki [26] and the SPAM algorithm proposed by Ayres et al. [27].

In addition, Yu et al. [28] used a Boolean matrix to improve the Apriori algorithm and implemented the algorithm on the Hadoop platform. Youcef et al. [29] proposed the GA Apriori and PSO Apriori algorithm by combining a genetic algorithm and particle swarm optimization algorithm. Du [30] proposed an improved Apriori algorithm based on a Boolean matrix and sort index rules. Anil vasoya et al. [31] improved the Apriori algorithm by combining distributed computing with parallel computing so as to find frequent itemsets from large databases in a shorter time. Agustinet al. [32] proposed COP algorithm based on the PrefixSpan algorithm to optimize the correlation of results obtained in the process of sequential pattern mining.

1.2. The Main Contribution of This Paper

At present, there is little research on status set sequential pattern mining with time windows, but this has great research value and significance in practical application. It can discover easily ignored rules in local windows and mine the periodicity of these rules.

The traditional sequential pattern mining methods only consider the ID of items in the temporal database, which ignores the relationship between items with different status. Additionally, the traditional sequential pattern mining methods only have the support constraint, which may lead to the mining rules still lacking application value. More

importantly, the traditional sequential pattern mining methods are to mine the temporal database over the whole time period and count the itemsets and sequences, while the sequential patterns existing in the local event window are often ignored. To solve the above problems, this paper improved the traditional sequential pattern mining methods.

Considering the status attribute of each item in the database for some complex systems and considering the status of the item, we can find the system between the normal state and the fault state at different time points, and then, according to the information mined, we can carry out preventive maintenance for the system that is about to have a fault state in advance.

The time window, confidence degree, coverage, factor set ratio and other constraints are introduced to mine the sequence patterns existing in local time windows, to find out the more regular sequence patterns among them and to mine the factor sets that lead to the occurrence of some key itemsets in the local time window.

The periodicity of the sequential pattern with the time window is analyzed. The law of periodic occurrence often has higher application value.

Through the improvement of traditional methods, this paper proposes the status set sequential pattern mining method with time window; it also improved the Apriori algorithm and proposes the TW-Apriori algorithm to mine the status set sequential pattern with time window.

1.3. The Main Content of This Paper

In the Section 2, the related concepts and constraints of status set sequential pattern mining with time windows are explained; in the Section 3, the main ideas of the TW-Apriori algorithm proposed in this paper are explained, and the algorithm is verified by a small-scale example; in the Section 4, the algorithm is verified by a large-scale example, and the results are analyzed; in the Section 5, the conclusion of this paper is given.

2. Problem Description for SSPMTW

2.1. Related Definitions and Properties

The following defines important concepts in the mining process of status set sequence patterns with time windows, where Table 1 is the symbol definition of the status set sequence pattern mining model with time windows.

Table 1. Symbol definitions for SSPMTW models.

Symbol	Definition
I	A collection of all status items, symbolized as $I = \{i_1, i_2, \dots, i_m\}$
X	A set of status items, such as X , can be represented as $X \subseteq I$
D	Collection of all events in a time series database
T	The entire time period in which the time series database is located
W	Collection of time windows
w	A time window, such as $w = [t_s, t_e]$, represents a continuous time interval that starts at t_s and ends at t_e
$ w $	Width of time window
D^w	Collection of events occurring in the w time period
$ D^w $	Number of events occurring in the w time period
$D(X)^w$	Collection of events occurring in the w time period that contain status itemset X
$ D(X)^w $	Number of events in $D(X)^w$
$X_1 \rightarrow^w X_2$	Sequence $X_1 \rightarrow X_2$ that appears in the w time window
$ X_1 \rightarrow X_2 ^w$	Number of occurrences of sequence $X_1 \rightarrow X_2$ in the w time window
$sup(X_1 \rightarrow^w X_2)$	Support for $X_1 \rightarrow^w X_2$ within the w time window can be expressed as $ X_1 \rightarrow X_2 ^w / D^w $
$c(X_1 \rightarrow^w X_2)$	Confidence for $X_1 \rightarrow^w X_2$ within the w time window can be expressed as $ X_1 \rightarrow X_2 ^w / D(X_1)^w $
$s\%$	User-specified minimal support, <i>minsup</i>
$c\%$	User-specified minimal confidence, <i>minconf</i>
ω	User-specified minimal width of time window, <i>minwin</i>
$g\%$	User-specified minimal time coverage rate, <i>mintcr</i>
$d\%$	User-specified minimal coverage rate, <i>mincov</i>
$u\%$	User-specified minimal factor set rate, <i>minfs</i>
$e\%$	User-specified minimal periodic time coverage rate, <i>minptcr</i>

Definition 1. Status Itemset (SI): An itemset can be defined as a status itemset if and only if it meets the following constraints:

1. $X \subseteq I$;
2. $\forall i \in X, i \in \{1, 2\}$;
3. $\forall j \in I - X, j = 0$.

Definition 2. Frequent Status Itemset with Time Windows (FSITW): FSITW can be expressed as X^W if and only if it meets the following constraints:

1. $X \subseteq I$;
2. $\forall w_i, w_j \in W, w_i \cap w_j = \emptyset$;
3. $\forall w \in W, \frac{|D(X)^w|}{|D^w|} \times 100\% \geq s\%$.

When mining frequent status itemsets with time windows, this paper draws on the downward closure property of the Apriori algorithm, puts forward the two concepts of subset and superset and proposes three new important properties based on these to calculate efficiency. This is shown below.

Definition 3. Subset: The set of status items within (t_s, t_e) the time window can be represented as $X^{(t_s, t_e)}$, if $X'^{(t_s, t_e)}$ is a subset of $X^{(t_s, t_e)}$, if and only if $X' \subseteq X, t'_s < t'_e, t'_s \geq t_s, t'_e \leq t_e$.

Definition 4. Superset: The set of status items within (t_s, t_e) the time window can be represented as $X^{(t_s, t_e)}$, if $X'^{(t_s, t_e)}$ is a superset of $X^{(t_s, t_e)}$, if and only if $X \subseteq X', t'_s < t'_e, t'_s \leq t_s, t'_e \geq t_e$.

Property 1. If status itemset $X(X \subseteq I)$ is frequent in the time window (t_s, t_e) , then any subset of it must also be frequent in the time window (t_s, t_e) .

Proof. Since the status itemset $X(X \subseteq I)$ is frequent in the time window (t_s, t_e) , $\frac{|D(X)^{(t_s, t_e)}|}{|D^{(t_s, t_e)}|} \times 100\% \geq s\%$. Since X' is a subset of X , $|D(X')^{(t_s, t_e)}| \geq |D(X)^{(t_s, t_e)}|$, $\frac{|D(X')^{(t_s, t_e)}|}{|D^{(t_s, t_e)}|} \times 100\% \geq s\%$. Thus, X' is also frequent in the time window (t_s, t_e) . \square

Property 2: If status itemset $X(X \subseteq I)$ is infrequent in the time window (t_s, t_e) , then any superset of it must also be infrequent in the time window (t_s, t_e) .

Proof. Since the status itemset $X(X \subseteq I)$ is infrequent in the time window (t_s, t_e) , $\frac{|D(X)^{(t_s, t_e)}|}{|D^{(t_s, t_e)}|} \times 100\% \leq s\%$. Since X' is a superset of X , $|D(X')^{(t_s, t_e)}| \leq |D(X)^{(t_s, t_e)}|$, $\frac{|D(X')^{(t_s, t_e)}|}{|D^{(t_s, t_e)}|} \times 100\% \leq s\%$. Thus, X' is also infrequent in the time window (t_s, t_e) . \square

Property 3. If status itemset $X(X \subseteq I)$ is infrequent in the time window W_1 , status itemset $Y(Y \subseteq I)$ is infrequent on W_2 , and $W_1 \cap W_2 = \emptyset \cap W_2 = \emptyset$, so the status itemset $Z = X \cup Y$ must be infrequent in the time window $W_3 = W_1 \cup W_2$.

Proof. Since the status itemset $X(X \subseteq I)$ is infrequent in the time window W_1 and the status itemset $Z = X \cup Y$ is a superset of X , according to property 2, Z is infrequent in the time window W_1 . Similarly, Z is infrequent in the time window W_2 , i.e., $\frac{|D(Z)^{W_1}|}{|D^{W_1}|} \times 100\% \leq s\%$, $\frac{|D(Z)^{W_2}|}{|D^{W_2}|} \times 100\% \leq s\%$. Assuming $|D^{W_1}|$ is greater than $|D^{W_2}|$, then $\frac{|D(Z)^{W_3}|}{|D^{W_3}|} \times 100\% \leq \frac{|D(Z)^{W_1}| + |D(Z)^{W_2}|}{2|D^{W_1}|} \times 100\% \leq s\%$. Thus, the status itemset $Z = X \cup Y$ is infrequent in the time window $W_3 = W_1 \cup W_2$. \square

These three important properties have an important value and role in frequent status set mining and can greatly reduce the number of candidate status itemsets, thus greatly improving the computational efficiency.

Definition 5. Status-Set Sequence with Time Windows (SSTW): SSTW is a collection of ordered status items with time windows such as $X \rightarrow_W Y$, $X, Y \subseteq FSITW$, and W represents the time window in which $X \rightarrow Y$ occurs.

There are also two important properties when searching for a candidate SSTW, which can greatly improve the speed of the algorithm, as shown in Properties 4 and 5 below.

Property 4. If the status set sequence $SS(SS \subseteq I)$ in the time window (t_s, t_e) is frequent, then any subset of its status set sequence must also be frequent in the time window (t_s, t_e) .

Proof. Since the status set sequence $SS(SS \subseteq I)$ is frequent in the time window (t_s, t_e) , $|SS|^{(t_s, t_e)} / |D^{(t_s, t_e)}| \times 100\% \geq s\%$. Since SS' is a subset of SS , $|SS'|^{(t_s, t_e)} \geq |SS|^{(t_s, t_e)}$, $|SS'|^{(t_s, t_e)} / |D^{(t_s, t_e)}| \times 100\% \geq s\%$. Thus, SS' is also frequent in the time window (t_s, t_e) . \square

Property 5. If the status set sequence $SS(SS \subseteq I)$ in the time window (t_s, t_e) is infrequent, then any of its super status set sequences must also be infrequent in the time window (t_s, t_e) .

Proof. Because the status set sequence $SS(SS \subseteq I)$ is infrequent in the time window (t_s, t_e) , $|SS|^{(t_s, t_e)} / |D^{(t_s, t_e)}| \times 100\% \leq s\%$. Since SS' is a superset of SS , $|SS'|^{(t_s, t_e)} \leq |SS|^{(t_s, t_e)}$, $|SS'|^{(t_s, t_e)} / |D^{(t_s, t_e)}| \times 100\% \leq s\%$. Thus, SS' is also infrequent in the time window (t_s, t_e) . \square

Definition 6. Mean Support of SSTW: The average support for SSTW of the form $X \rightarrow_W Y$ can be defined as $\bar{s}\% = (\sum_{w \in W} |w| \cdot s_w) / (\sum_{w \in W} |w|) \times 100\%$, where s_w is the support of $X \rightarrow_W Y$, i.e., $s_w = (|D(X \rightarrow Y)^w|) / (|D^w|), \forall w \in W$.

Definition 7. Mean Confidence of SSTW: The average confidence for SSTW of the form $X \rightarrow_W Y$ can be defined as $\bar{c}\% = (\sum_{w \in W} |w| \cdot c_w) / (\sum_{w \in W} |w|) \times 100\%$, where c_w is the confidence of $X \rightarrow_W Y$, i.e., $c_w = (|D(X \rightarrow Y)^w|) / (|D(X)^w|), \forall w \in W$.

Definition 8. Time Coverage Rate of SSPTW: The time coverage of $X \rightarrow_W Y$, $tcr\%$, indicates the coverage of the SSPTW over the entire time period, which can be expressed as follows:

$$tcr\% = \frac{\sum_{w \in W} |w|}{|T|} \times 100\% \quad (1)$$

where $\sum_{w \in W} |w|$ is the total length of the time window in which $X \rightarrow_W Y$ resides, $|T|$ represents the total time interval of the time series database. In special cases, when $tcr\% = 100\%$, the SSPTW is a traditional, full-time sequence pattern; when $tcr\% \leq 100\%$, the SSPTW is a part-time sequence pattern. Since the width of the minimum time window ω is set in this paper and the following research will also be based on the minimum time window, $tcr\%$ must satisfy the following restrictions: $tcr\% \geq \omega / T$.

Definition 9. Status Set Sequential Pattern with Time Windows (SSPTW): $X \rightarrow_W Y$ is SSPTW, if and only if it meets the following constraints:

1. $X \subseteq SI, Y \subseteq SI, X \cap Y = \emptyset, W \neq \emptyset$;
2. $X^W, Y^W \subseteq FSITWs$, that is, X^W and Y^W meet the user-defined minimum support $s\%$;
3. $\forall w \in W, (|D(X \rightarrow Y)^w|) / (|D^w|) \times 100\% \geq s\%$;

4. $\forall w \in W, (|D(X \rightarrow Y)^w|) / (|D(X)^w|) \times 100\% \geq c\%$;
5. $\forall w \in W, |w| \geq \omega$.

Based on the above definition, it is known that an SSPTW's average support is always not less than the minimum support, and its average confidence is not less than the minimum confidence; otherwise, a sequence with large average support and average confidence cannot ensure that it is an SSPTW.

Definition 10. Coverage Rate (CR) of SSPTW: The coverage of the status set sequence, $SSPTW = \{X_1 \rightarrow X_2 \rightarrow \dots \rightarrow X_k\}^W$, can be expressed as:

$$CR(SSPTW) = \min \left\{ \frac{|X_1 \rightarrow X_2 \rightarrow X_3 \rightarrow \dots \rightarrow X_k|^W}{|X_k|^W} \times 100\% \mid \forall k = 2, 3, \dots, p \right\} \quad (2)$$

Definition 11. Strong Status Set Sequential Pattern with Time Windows (Strong SSPTW): The status set sequence pattern (SSPTW) is a strong status set sequence pattern if and only if $CR(SSPTW) \geq \text{mincov}$, where mincov is the user-defined minimum coverage threshold ($d\%$), which is assigned the same value as the minimum confidence threshold ($c\%$).

Definition 12. Factor Set of FSI (FS): The factor set of status itemset X can be expressed as $FS(X)$, for any status set sequence pattern ssp in $\{ssp_1, ssp_2, \dots, ssp_i\}$; if $ssp \rightarrow_W X$ is still a status set sequence pattern, then ssp is an element of $FS(X)$. The factor set of the frequent status itemset X can be expressed as:

$$\text{Rate}(X) = \sum_i (CR(ssp_i \rightarrow_W X)) \quad (3)$$

The ssp_i is an element in factor set $FS(X)$, and i is the subscript of each element.

When it satisfies Formula (2.4), where minfs is the user-defined minimum factor set ratio $u\%$, $FS(X)$ is called the main factor set of frequent status set X .

$$\text{Rate}(X) = \sum_i (CR(ssp_i \rightarrow_W X)) \geq \text{minfs} \quad (4)$$

2.2. Periodic Analysis of SSPTW

A periodic status set sequence pattern is a collection of sequence patterns and a series of time windows, such as $X \rightarrow_{P_i} Y$, where P_i is a periodic time window containing specific events. Key concepts in the mining process of periodic status set sequential patterns are defined below.

Definition 13. Periodic Width, T: This is the user-specified width of a time window that satisfies the periodic pattern, where the periodic width is an integer multiple of the minimum time window, that is, $T = k \times \omega$.

Definition 14. Periodic Interval, O: This is a user-defined time window interval that satisfies a periodic pattern, where the periodic interval is an integer multiple of the minimum time window, that is, $O = k \times \omega$, where the periodic interval can be determined by experience by day, week, month, year, etc.

Definition 15. Periodic SSP: A status set sequence pattern, such as $X \rightarrow Y$, when both the periodic width T and periodic interval O are satisfied. The status set sequence pattern is the periodic status set sequence pattern, which is expressed as $X \rightarrow_{T,O} Y$.

Definition 16. Periodic Time Coverage Rate (PTCR): Assume that p is the number of cycles for a time window that satisfies a periodic SSP and $n-p$ is the number of cycles for a time window

that does not meet that periodic SSP. $X \rightarrow_{T,O} Y$ is a periodic SSP with strong regularity if and only if $X \rightarrow_{T,O} Y$ satisfies the following periodic time coverage values:

$$\frac{p}{n - p} \times 100\% \geq e\% \tag{5}$$

Among them, $e\%$ is the user-defined minimum periodic time coverage threshold. When a periodic pattern $X \rightarrow_{P_i} Y$ satisfies both T and O constraint thresholds, but $PTCR(X \rightarrow_{P_i} Y) < e\%$, this is a periodic SSP with weak periodic time coverage rules.

Definition 17. Periodic Analysis of Patterns: Periodic analysis of patterns refers to the mining of repeated status set sequence patterns with periodic regularity in a time series database.

Figure 1 is a framework for SSPMTW. In this paper, the sequential pattern mining model is extended to mine SSP with time windows, strong SSP with time windows, the main factor set of a frequent status itemset with time windows and periodic sequential patterns.

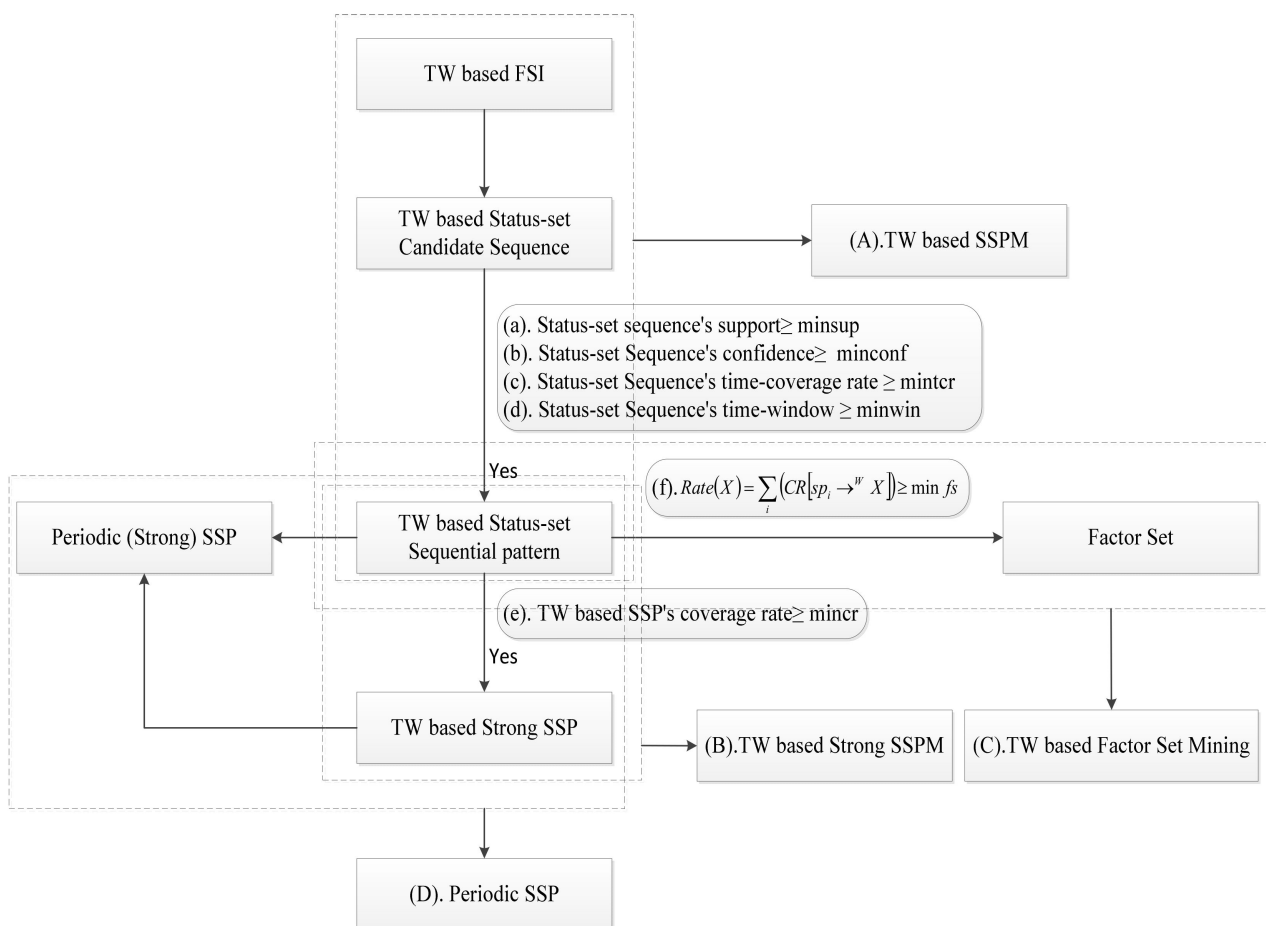


Figure 1. Frame diagram of SSPM with time window.

3. SSPTW Mining Algorithm

3.1. Mining Large-One FSITWs

Mining the largest set of frequent one-status items with time windows (large-one FSITWs) are the most important part of status set sequence pattern mining with time windows because these are the basis for producing n-FSITWs and SSPTW. Since the time window of FSITW must be no less than the user-defined minimum time window threshold of minwin, a simple method of mining large-one FSITWs can take the following two steps:

- (1) Divide the time series database into several time periods through minwin;
- (2) Count the support degree of individual items in each time period of the database and compare with the user-defined minimum support degree to determine whether the set of status items are frequent. If the set of status items are frequent in one time window and not frequent in adjacent time windows, the two time windows are merged, and the support degree of the set of status items on the merged time window is counted. If the minimum support threshold is still met, the set of status items are considered frequent in the merged time window.

Based on this idea, this paper maximizes the time window of the frequent one-status itemset in preparation for finding frequent n-FSITWs in the next step. The time interval for frequent time windows obtained by this method will be an integer multiple of minwin.

3.2. Research on SSPMTW Mining Algorithm

The SSPTW mining steps are as follows:

- (A) Mining a set of frequent status items with time windows, FSITW.
- (B) Mining the status set sequence pattern with time windows, SSPTW.
- (C) Mining periodic status set sequence patterns with time windows, periodic SSPTW.

3.2.1. FSITW Algorithm

This paper proposes the TW-Apriori algorithm to mine SSPMTW and analyze its periodicity. The main idea of the algorithm is introduced below.

The traditional sequential pattern mining algorithm is to mine frequent itemsets in the whole temporal database, ignoring the frequent itemsets in the local time window. The frequent itemsets are the basis of mining sequential patterns, so the traditional sequential pattern mining algorithms cannot mine the sequential patterns in the local time window. Therefore, mining frequent status itemsets FSITWs in local time window is the key.

Generating large-one FSITWs is the basis of mining frequent status itemsets with time windows. This algorithm divides the database into small segments according to the minwin threshold, then scans the whole database, counts the support of one-candidate status itemsets in each small time segment, compares them with minsup to determine whether they are frequent, and uses the idea described in Section 3.1 to maximize the time window of one-candidate status itemsets by combining the adjacent time window. If the time coverage rate of the time window where the one-candidate status itemset is located meets mintcr, then the one-candidate status itemset is large-one FSITWs.

The specific pseudo code is shown in Figure 2. The inputs are the temporal database, minwin, minsup and mintcr, and the outputs are large-one FSITWs.

When mining FSITWs, we need to generate candidate k-FSITWs ($k \geq 2$). The idea of the algorithm is to generate the next candidate k-FSITWs through the (k-1)-FSITWs mined in the previous step until no candidate status itemset is generated. It is worth mentioning that for all candidate status itemsets, only when their time window is in the intersection of their time window supersets can we count their support, which is also the core idea of property 3 in Section 2.

The specific pseudo code is shown in Figure 3. The inputs are the (k-1)-FSITWs mined in the previous step and minsup, and the outputs are the candidate k-FSITWs.

```

Subroutine: Large1_gen()
Input:  $D, minwin, minsup, mintcr$ 
Output:  $L_1$ 
Begin
(1)  $D = \cup D_{T_i}, |T_i| = |(t_s, t_e)| = minwin$ 
(2)  $L_1 = \emptyset$ 
(3) For all individual item  $x \in SI$  do begin
(4)    $T = \emptyset$ 
(5)    $t = \emptyset$ 
(6)    $C_1 = \emptyset$ 
(7)   for  $i = 1:n$  do begin
(8)      $t = t \cup T_i$ 
(9)     Count support of  $x$  in  $D_t$ 
(10)    If  $(support(D_t(x)) \geq minsup)$  then
(11)       $T = T \cup T_i$ 
(12)    Else
(13)       $C_1 = C_1 + x^T$ 
(14)       $T = \emptyset$ 
(15)       $t = \emptyset$ 
(16)    End if
(17)  End for
(18)  If  $tcr(C_1) \geq mintcr$  then
(19)     $L_1 = L_1 \cup C_1$ 
(20)  End if
(21) End for
(20)  $Output = L_1$ 
End

```

Figure 2. Large-one FSITWs mining algorithm.

```

Subroutine: Candidate_gen()
Input:  $L_{k-1}, minsup$ 
Output:  $C_k$ 
Begin
(1)  $C_k = \emptyset$ 
(2) For all  $x \in SI$  do begin
(3)   For all pairs of  $l_i^{[t_s, t_e]}, l_j^{[t_s, t_e]} \in L_{k-1}$  do begin
(4)     Candidate =  $l_i \cap l_j$ 
(5)      $T = [t_s, t_e]_i \cap [t_s, t_e]_j$ 
(6)     Count support of Candidate in  $T$ 
(7)     If  $|Candidate| = k \wedge T \neq \emptyset \wedge support \geq minsup$  then
(8)       Put candidate with  $T$  into  $C_k$ 
(9)     End if
(10)  End for
(11) End for
(12)  $Output = C_k$ 
End

```

Figure 3. New candidate FSITWs mining algorithm.

Based on the above two subroutines, this paper provides the pseudo code for mining FSITWs, as shown in Figure 4. First, large-one FSITWs are generated, and then, all candidate k-FSITWs are generated by using the subroutine shown in Figure 3, and their support and time coverage rate are counted. When the minimum support threshold and minimum time coverage rate threshold are met, FSITWs can be obtained. In this paper, we will use the properties 1–3 proposed in the second section to reduce the generation of candidate itemsets so as to improve the efficiency of the algorithm.

```

Main: TW-Apriori Algorithm on Finding FSITWs ()
Input:  $D, minwin, minsup, mintcr$ 
Output: set of FSITW
Begin
(1)  $L_1 = Large_1\_gen(D, minsup)$ 
(2) for ( $l = 2; L_{l-1} \neq \emptyset; l++$ ) do
(3)  $C_l = Candidate\_gen(L_{l-1})$ 
(4)   For each  $C_l = Candidate\_gen(L_{l-1})$  do
(5)     Count support of  $c$  in its frequent-possible time windows
(6)       If  $c \cdot support \geq minsup$ 
(7)          $tcr(c) \geq mintcr$  then
(8)            $L_l = L_l \cup \{c\}$ 
(9)         End if
(10)    End for
(11) End for
(12)  $Output = \cup L_l$ 
End

```

Figure 4. TW-Apriori algorithm for mining FSITWs.

Using the above three algorithms, we finally mined all the frequent status itemsets and their time windows to prepare for the next sequential pattern with time windows.

3.2.2. SSPTW Algorithm

All FSITWs have been mined in Section 3.3.1, which are the basis of mining status set sequential patterns with time windows. In the process of mining SSPTW, we need to combine (k-1)-SSPTW ($k \geq 2$) and 1-SSPTW to generate candidate k-SSPTW, in which 1-SSPTW is all FSITWs. The specific pseudo code is shown in Figure 5. Its inputs are the (k-1)-SSPTW mined in the previous step and FSITWs, and its outputs are the candidate k-SSPTW.

```

Subroutine: Candidate_gen()
Input:  $SSPTW_{l-1}, FSITW$ 
Output:  $Cand-SSPTW_l$ 
Begin
(1)  $Cand - SSPTW_l = \emptyset$ 
(2) For all  $SSPTW_i \in SSPTW_{l-1}$  do begin
(3)   For all  $y_j^W \in FSITW$  do begin
(4)      $Cand = SSPTW_i \cap y_j^W$ 
(5)      $T = [t_s, t_e]_{SSPTW_i} \cap [t_s, t_e]_{y_j^W}$ 
(6)     Count support  $Cand$  in  $T$ 
(7)     If  $|Cand| = k$  and  $T \neq \emptyset$  and  $support_{cand} \geq minsup$  then
(8)       Put  $Cand$  with  $T$  into  $Cand-SSPTW_l$ 
(9)     End if
(10)  End for
(11) End for
(12)  $Output = Cand - SSPTW_l$ 
End

```

Figure 5. New Candidate-SSPTW mining algorithm.

Using the above subroutine, we can obtain all candidate SSPTWs, and calculate their support, confidence and time coverage rate so as to mine SSPTW. Figure 6 shows the specific pseudo code of the SSPTW mining algorithm.

```

Main: TW-Apriori Algorithm on discovering SSPTW()
Input: FSITW, minsup, minconf, mintcr
Output: SSPTW
Begin
(1)  $L_0 = FSITW$ 
(2) for ( $l = 1; L_{l-1} \neq \emptyset; l++$ ) do
(3)    $SSPTW_l = \emptyset$ 
(4)    $C_l = candidate\_gen(L_{l-1}, L_0)$ 
(5)   For each candidate  $m \in C_l$  do
(6)     Calculate support of  $m$  in its frequent-possible time-windows
(7)     If  $Sup_m \geq minsup$ 
(8)        $Conf_m \geq minconf$ 
(9)        $TCR_m \geq mintcr$  then
(10)         $SSPTW_l = SSPTW_l \cup \{m\}$ 
(11)     End if
(12) End for
(13) End for
(14)  $Output = \cup SSPTW_l$ 
End

```

Figure 6. TW-Apriori algorithm for mining SSPTW.

3.2.3. Strong SSPTW Algorithm

After the study in Section 3.2.2, all the status set sequence patterns with time windows will be finally discovered. In order to solve the problem that the sequence pattern mined by the traditional algorithm has weak regularity and low practical application value, this paper adds the constraint of coverage rate to mine SSPTW with stronger rules, that is, strong SSPTW. Figure 7 is the specific pseudo code of the strong SSPTW mining algorithm. By counting the coverage of SSPTW and judging whether it meets the coverage threshold, the strong SSPTW is mined.

```

Main: TW-Apriori Algorithm on discovering Strong SSPTW()
Input:  $SSPTW_l, mincov$ 
Output: Strong SSPTW
Begin
(1) for ( $l = 1; L_{l-1} \neq \emptyset; l++$ ) do
(2)    $L_l = SSPTW_l$ 
(3)    $Strong\ SSPTW_l = \emptyset$ 
(4)   For each  $h \in L_l$  do
(5)     Calculate its coverage rate  $CR(h)$ 
(6)     If  $CR(h) \geq mincov$  then
(7)        $Strong\ SSPTW_l = Strong\ SSPTW_l \cup \{h\}$ 
(8)     End if
(9)   End for
(10) End for
(11)  $Output = \cup Strong\ SSPTW_l$ 
End

```

Figure 7. TW-Apriori algorithm for mining Strong SSP.

3.2.4. Factor Set of FSITW Algorithm

In the practical application process, users may be interested in some important FSITWs, because when these important FSITWs occur, they may bring huge economic benefits or

huge losses, or other important impacts. Therefore, the constraint of the minimum factor set ratio is added in this paper.

Figure 8 shows the specific pseudo code of the factor set of the FSITWs mining algorithm. According to definition 12 in Section 2, when the factor set of the FSITWs meets the minimum factor set ratio, it is the main factor set of the FSITWs.

```

Main: TW-Apriori Algorithm on discovering FS()
Input: FSI, SSP, minfs
Output: Factor set of FSI
Begin
(1)  $L_0 = FSI^W$ 
(2) For each  $y \in FSI^W$  do
(3)    $FS(y) = \emptyset$ 
(4)   for ( $l = 1; L_{l-1} \neq \emptyset; l++$ ) do
(5)     For each  $ssp_l^w \in SSP_l^W$ 
(6)       If  $Rate(y) = \sum_i (CR(ssp_i \rightarrow^w y)) \geq minfs$  then
(7)          $FS(y) = FS(y) \cup \{ssp_l^w\}$ 
(8)       End if
(9)     End for
(10)  End for
(11) End for
(12)  $Output = FS(y)$ 
End

```

Figure 8. TW-Apriori algorithm for mining the FSI factor set.

3.2.5. Periodic SSP Algorithm

By analyzing the periodicity of the status set sequence pattern with time windows, the regularity of the periodicity in the pattern is found. Firstly, the SSPTW and its time windows satisfying the period width T and period interval O are discovered, and then, the periodic time coverage rate of these patterns is calculated. When they meet the minimum periodic coverage rate threshold, the pattern is determined as a periodic status set sequence pattern.

The specific pseudo code of the algorithm idea is shown in Figure 9.

```

Main: TW-Apriori Algorithm on discovering Periodic SSPTW()
Input: SSPTWl, T, O, minptcr
Output: Periodic SSPTW
Begin
(1) for ( $l = 1; L_{l-1} \neq \emptyset; l++$ ) do
(2)    $L_l = SSPTW_l$ 
(3)    $Periodic\ SSPTW_l = \emptyset$ 
(4)   For each  $h \in L_l$  do
(5)     Match its time-windows with T and O
(6)     If  $T(h) \geq T, \text{ and } O(h) \geq O$  then
(7)       Calculate its periodic time coverage rate
(8)       If  $ptcr(h) \geq minptcr$  then
(9)          $Periodic\ SSPTW_l = Periodic\ SSPTW_l \cup \{h\}$ 
(10)      End if
(11)    End if
(12)  End for
(13) End for
(14)  $Output = \cup Periodic\ SSPTW_l$ 
End

```

Figure 9. Periodic sequence pattern mining algorithm.

3.2.6. Analysis of Computational Complexity of the Algorithm

When mining frequent itemsets with time windows, candidate frequent itemsets are obtained by pairwise combination of the previous frequent itemsets; this is also the case for the mining of status sets sequential patterns with time windows. This will generate a large number of candidate itemsets and sequences. For example, when n $(k-1)$ -FSITWs are mined in the previous step, C_n^2 candidate k -FSITWs will be generated. This will greatly occupy the memory of the computer and increase the computational complexity of the algorithm. The properties 1 to 5 proposed in the second section can greatly reduce the generation of candidate frequent itemsets and status sets sequential patterns, reducing the complexity and improving the efficiency of the algorithm.

3.2.7. Entropy in the Systems

Entropy is ubiquitous in the system, and its physical meaning is a measure of the degree of system chaos. The method proposed in this paper is to mine the potential rules in the system, so as to predict the future state of the system. These rules are sequential patterns, which reflect the relationship between different system states. The appearance of one state of the system may cause the occurrence of another state. Through these sequential patterns, we can know the future state of the system, so as to optimize the system. Therefore, the discovery of these sequential patterns can make the system more orderly and stable, and correspondingly, the entropy of the system decreases.

3.3. Solution and Analysis of an Example of SSPMTW

The following is a small-scale example to verify the feasibility and validity of the methods and algorithms presented in this paper.

Table 2 shows 25 instances of fault monitoring for three parts of a system, recording the corresponding data in chronological order and finally forming the time series database. The TID represents the timestamp of the monitoring; the Status itemset represents the set of status items for each monitoring record; the three parts correspond to three items— $\{i_1, i_2, i_3\}$, $status \in \{0, 1, 2\}$, 0—normal, 1—potential failure, 2—failure. $minwin = 5$, $minsup = 40\%$, $minconf = 60\%$, $mincov = 60\%$, $mintcr = 40\%$, $minfs = 80\%$.

Table 2. Temporal database.

TID	Status Itemset	TID	Status Itemset	TID	Status Itemset
1	$(i_1,1), (i_2,2), (i_3,2)$	10	$(i_1,2), (i_2,2)$	19	$(i_2,1), (i_3,2)$
2	$(i_1,2), (i_2,1), (i_3,1)$	11	$(i_1,1), (i_3,2)$	20	$(i_2,1)$
3	$(i_1,1), (i_2,2), (i_3,2)$	12	$(i_1,2), (i_2,1), (i_3,1)$	21	$(i_1,1), (i_3,2)$
4	$(i_1,2), (i_2,1), (i_3,1)$	13	$(i_1,1), (i_2,2)$	22	$(i_1,2), (i_2,1), (i_3,1)$
5	$(i_1,1), (i_2,2)$	14	$(i_1,2), (i_2,1), (i_3,1)$	23	$(i_1,1), (i_2,2)$
6	$(i_1,2), (i_2,1)$	15	$(i_2,2)$	24	$(i_1,2), (i_2,1), (i_3,1)$
7	$(i_3,2)$	16	$(i_1,2), (i_2,2)$	25	$(i_2,2)$
8	$(i_1,2), (i_3,2)$	17	$(i_1,1), (i_3,2)$		
9	$(i_1,1), (i_3,2)$	18	$(i_1,2), (i_3,1)$		

3.3.1. Solution and Analysis of SSPMTW

Step 1. Mining FSITW

This section uses two methods to mine frequent status itemsets. The first method is FSI mining without considering time windows. The second method is mining frequent status itemsets with time windows, that is, the FSITW mining method in this section. Using two different solutions, the two methods are compared and analyzed.

First, in order to easily calculate the support of each item, the temporal database needs to be converted to a 0–1 form, as shown in Table 3.

Table 3. Temporal database (0–1).

TID	Status Item						TID	Status Item					
	(i _{1,1})	(i _{1,2})	(i _{2,1})	(i _{2,2})	(i _{3,1})	(i _{3,2})		(i _{1,1})	(i _{1,2})	(i _{2,1})	(i _{2,2})	(i _{3,1})	(i _{3,2})
1	1	0	0	1	0	1	14	0	1	1	0	1	0
2	0	1	1	0	1	0	15	0	0	0	1	0	0
3	1	0	0	1	0	1	16	0	1	0	1	0	0
4	0	1	1	0	1	0	17	1	0	0	0	0	1
5	1	0	0	1	0	0	18	0	1	0	0	1	0
6	0	1	1	0	0	0	19	0	0	1	0	0	1
7	0	0	0	0	0	1	20	0	1	0	0	0	0
8	0	1	0	0	0	1	21	1	0	0	0	0	1
9	1	0	0	0	0	1	22	0	1	1	0	1	0
10	0	1	0	1	0	0	23	1	0	0	1	0	0
11	1	0	0	0	0	1	24	0	1	1	0	1	0
12	0	1	1	0	1	0	25	0	0	0	1	0	0
13	1	0	0	1	0	0	Sum	9	12	8	9	7	9

- (1) FSI Mining FSI Mining takes place over the entire time period of the database. If the support of a status itemset is not less than the minimum support specified by the user, it is determined that the set is a frequent status itemset. By summing the columns of each status item in Table 3, the support of each status item is calculated (as shown in red in the last line of Table 3). Since $\text{misup} = 40\%$, the minimum amount of support is $25 \times 40\% = 10$, which means that when the amount of support for each item is not less than 10, the item is frequent. As can be seen from the red letters in the last row of Table 3, only $(i_{1,2})$ of the status items meets the requirements, and all other items have less than 10 support, so the set of frequent status items finally mined by the FSI Mining method is $\{(i_{1,2})\}$.
- (2) Frequent status Item Set Mining with Time Window Frequent status itemsets with time windows are divided into small time periods by minwin , and then, the support of itemsets is determined to be minsup -satisfied in a small time period. When minsup is satisfied, the time coverage of these time windows is determined to be no less than mintcr . Only when both conditions are satisfied can the itemset be determined to be FSITW.
 - (A) 1-FSITW Figure 10 takes two status items, $(i_{1,1})$ and $(i_{3,2})$, as examples for analysis. The graphics show that $(i_{1,1})$ is frequent in time windows $[1,3,5]$ and infrequent in time windows $[2,4]$; $(i_{3,2})$ is frequent in time windows $[1,2,4]$ and infrequent in time windows $[3,5]$.
The time window maximization of frequent one-status itemsets means that, if the minsup constraint threshold is still satisfied after merging the frequent time window and the adjacent infrequent time window, we will merge their time windows. $(i_{1,1})$ is frequent in time windows $[1,3,5]$ but is not frequent in time windows $[2,4]$. However, by maximizing the time window of frequent one-status itemsets, we find that $(i_{1,1})$ is also frequent in time windows $[1-3]$, and its support is $6/15$, which meets minsup . Then, by considering the minimum time coverage threshold, we find that $\text{tcr}(i_{1,1}) = 3/5 > 40\%$, so the maximum frequent time windows of $(i_{1,1})$ are $[1-3,5]$. Similarly, using the same method, we find that the frequent time windows of $(i_{3,1})$ are $[1-4]$. Summarily, using the same method, we mined the frequent one-status itemsets with the largest time window, namely 1-MFSITW.
 - (B) k-FSITW Based on the 1-FSITW, this paper finally mines all FSITWs, and makes a simple comparison between FSITW and FSI, as shown in Table 4.

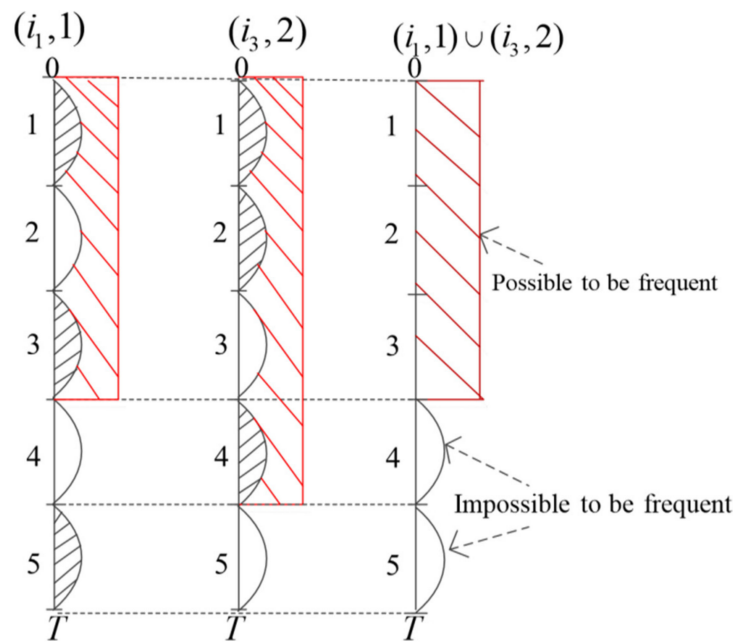


Figure 10. An example illustrating FSITW.

Table 4. Comparison between FSITW and FSI.

	Status Itemset	FSITW			FSI
		Time Window	Support	Time Coverage Rate	Status Itemset
L ₁	(i ₁ ,1)	[1–3,5]	[0.4,0.4]	0.8	(i ₁ ,2)
	(i ₁ ,2)	[1–5]	[0.48]	1.0	
	(i ₂ ,1)	[1,3,5]	[0.4,0.4,0.4]	0.6	
	(i ₂ ,2)	[1–3,5]	[0.4,0.4]	0.8	
	(i ₃ ,1)	[1,3,5]	[0.4,0.4,0.4]	0.6	
	(i ₃ ,2)	[1–4]	[0.4]	0.8	
L ₂	(i ₁ ,2)(i ₂ ,1)	[1,3,5]	[0.4,0.4,0.4]	0.6	∅
	(i ₁ ,2)(i ₃ ,1)	[1,3,5]	[0.4,0.4,0.4]	0.6	
	(i ₂ ,1)(i ₃ ,1)	[1,3,5]	[0.4,0.4,0.4]	0.6	
L ₃	(i ₁ ,2)(i ₂ ,1)(i ₃ ,1)	[1,3,5]	[0.4,0.4,0.4]	0.6	∅
Number		10			1

It can be seen from the table that 10 FSITWs satisfying the conditions can be mined in the local time window, while only one FSI can be mined in the whole time period. Through comparison, it can be found that the proposed method can mine those status itemsets that are not frequent in the whole time period but are frequent in the local time window.

Step 2. Mining SSPTWs

When mining sequential patterns of status sets with time windows, we need to consider the constraints of support, confidence and time coverage. Only when the sequence meets the threshold of minsup, minconf and mintcr can it be judged as sequential patterns of status sets with time windows. Among them, minsup = 40%, minconf = 60%, mintcr = 40%. The specific SSPTW is shown in Table 5. In the following table, L₁ represents an SSPTW with a length of 1, that is, all FSITWS, and L_k represents the k-status set sequence pattern with time window.

Table 5. Status-set sequential pattern with time window.

	SSPTW	TW	Support	Confidence	Coverage Rate	TCR
L1	$(i_{1,1}) \dots (i_{1,2})(i_{2,1})(i_{3,1})$	*	*	*	*	*
L2	$(i_{1,1}) \rightarrow (i_{1,2})$	[1,3,5]	[0.4,0.4,0.4]	[0.67,1.0,1.0]	[1.0,1.0,1.0]	0.6
	$(i_{1,1}) \rightarrow (i_{2,1})$	[1,3,5]	[0.4,0.4,0.4]	[0.67,1.0,1.0]	[1.0,1.0,1.0]	0.6

	$(i_{2,1})(i_{3,1}) \rightarrow (i_{2,2})$	[1,3,5]	[0.4,0.4,0.4]	[1.0,1.0,1.0]	[0.67,1.0,1.0]	0.6
L3	$(i_{1,2})(i_{2,1})(i_{3,1}) \rightarrow (i_{2,2})$	[1,3,5]	[0.4,0.4,0.4]	[1.0,1.0,1.0]	[0.67,1.0,1.0]	0.6
	$(i_{1,1}) \rightarrow (i_{1,2}) \rightarrow (i_{2,2})$	[1,3,5]	[0.4,0.4,0.4]	[1.0,1.0,1.0]	[0.67,1.0,1.0]	0.6
	$(i_{1,1}) \rightarrow (i_{2,1}) \rightarrow (i_{2,2})$	[1,3,5]	[0.4,0.4,0.4]	[0.67,1.0,1.0]	[0.67,1.0,1.0]	0.6

	$(i_{1,1}) \rightarrow (i_{2,1})(i_{3,1}) \rightarrow (i_{2,2})$	[1,3,5]	[0.4,0.4,0.4]	[0.67,1.0,1.0]	[0.67,1.0,1.0]	0.6
	$(i_{1,1}) \rightarrow (i_{1,2})(i_{2,1})(i_{3,1}) \rightarrow (i_{2,2})$	[1,3,5]	[0.4,0.4,0.4]	[0.67,1.0,1.0]	[0.67,1.0,1.0]	0.6

Step 3. Mining Strong SSPTWs

On the basis of SSPTWs, strong SSPTWs can be mined by considering the coverage threshold, where mincov = 60%.

It can be seen from the “coverage rate” column in Table 5 that all SSPTWs meet the mincov constraint, that is, SSPTWs in Table 5 are strong SSPTWs.

Step 4. Mining Factor Set of FSITWs

When considering the factor set rate constraint, we can mine the major factor set of FSITW with time windows. We took frequent status itemsets with time windows $\{(i_{2,2})-[1-3,5]\}$ as an example, in which we assumed that $(i_{2,2})$ was an important part that users pay close attention to. Finally, we discovered the major factor set of $(i_{2,2})$, as shown in Table 6.

Table 6. Major factor set of FSITW.

FSI	L _k	Major Factor Set	TW	Support	Confidence	TCR
$(i_{2,2})-[1-3,5]$	L ₁	$(i_{1,2})$	[1,3,5]	[0.4,0.4,0.4]	[1.0,1.0,1.0]	0.6
		$(i_{2,1})$	[1,3,5]	[0.4,0.4,0.4]	[1.0,1.0,1.0]	0.6
	
	L ₂	$(i_{1,2})(i_{2,1})(i_{3,1})$	[1,3,5]	[0.4,0.4,0.4]	[1.0,1.0,1.0]	0.6
		$(i_{1,1}) \rightarrow (i_{1,2})$	[1,3,5]	[0.4,0.4,0.4]	[0.67,1.0,1.0]	0.6
		$(i_{1,1}) \rightarrow (i_{2,1})$	[1,3,5]	[0.4,0.4,0.4]	[0.67,1.0,1.0]	0.6
	
		$(i_{1,1}) \rightarrow (i_{1,2})(i_{2,1})(i_{3,1})$	[1,3,5]	[0.4,0.4,0.4]	[0.67,1.0,1.0]	0.6

In Table 6, we can find all the major factor sets that may lead to $(i_{2,2})-[1-3,5]$. At the same time, we sorted the support, confidence and time coverage of each SSPTW that led to $(i_{2,2})-[1-3,5]$ from large to small so that we could quickly lock high-frequency and effective SSPTWs, which is convenient for users to focus on monitoring these sequential patterns.

3.3.2. Periodic Analysis of SSPTW

Using the periodic analysis of SSPTWs in Table 5, we finally found periodic SSPs, as shown in Table 7.

Table 7. Periodic SSPs.

TW	L _k	Periodic SSP	Support	Confidence	Coverage Rate	TCR
[1,3,5]	L ₂	(i _{1,2})→(i _{2,2})	[0.4,0.4,0.4]	[1.0,1.0,1.0]	[1.0,1.0,1.0]	0.6
		(i _{2,1})→(i _{2,2})	[0.4,0.4,0.4]	[1.0,1.0,1.0]	[0.67,1.0,1.0]	0.6
	
	(i _{1,1})→(i _{1,2})(i _{2,1})(i _{3,1})	[0.4,0.4,0.4]	[0.67,1.0,1.0]	[1.0,1.0,1.0]	0.6	
	L ₃	(i _{1,1})→(i _{2,1})→(i _{2,2})	[0.4,0.4,0.4]	[0.67,1.0,1.0]	[0.67,1.0,1.0]	0.6
		(i _{1,1})→(i _{3,1})→(i _{2,2})	[0.4,0.4,0.4]	[0.67,1.0,1.0]	[0.67,1.0,1.0]	0.6
...		
(i _{1,1})→(i _{1,2})(i _{2,1})(i _{3,1})→(i _{2,2})	[0.4,0.4,0.4]	[0.67,1.0,1.0]	[0.67,1.0,1.0]	0.6		
[2,4]	L ₂	(i _{3,2})→(i _{1,2})	[0.4,0.4,0.4]	[1.0,0.67,1.0]	[1.0,0.67,0.67]	0.6
		(i _{1,2})→(i _{3,2})	[0.4,0.4]	[0.67,0.67]	[0.67,1.0]	0.4

It can be seen from Table 7 that there are two periodic laws in the table: the periodic time windows are [1,3,5], [2,4] and the cycle width of these two windows is 1; the cycle interval is also 1, and the periodic time coverage is 100% and 66.7%, respectively.

Table 7 not only excavates periodic SSP but also sorts them according to support, confidence, coverage rate and TCR, so that users can select relatively high value periodic status set sequence patterns.

4. Large-Scale Example Experiment

This section will use large-scale examples to verify the feasibility and efficiency of the methods and algorithms proposed in the article. The data used in this section include 96,554 sets of transaction data composed of 1000 items. The description of the dataset is shown in Table 8.

Table 8. Dataset description.

Tid	Status Itemset
Timestamp of each transaction	The status itemset contained in each transaction

4.1. Analysis of Solution Results

4.1.1. TW-Apriori Algorithm Efficiency Verification

It can be seen from the curve trend in Figure 11 that the total running time increases with the increase in data size, but it still shows a linear growth trend rather than an exponential growth. Moreover, when the data scale is 90,000, the running time to finally mine all modes is only 11 s. Therefore, the curve trend in Figure 11 can prove the efficiency of the TW-Apriori algorithm proposed in this paper, where minwin = 5000, minsup = 0.035, minconf = 0.04, mintcr = 0.1.

4.1.2. Analysis of FSITW Mining Results

First, this paper will study the division of time window so as to mine the optimal minwin, as shown in Figure 12. By analyzing the three curves with minsup values of 0.030, 0.035 and 0.040, it was found that when minsup = 0.030, the number of FSITW decreased with the increase in minwin threshold. For the other two curves, it was found that the number of FSITW also showed a decreasing trend, but when the value of minwin was greater than 70,000, the two curves basically coincided; that is, when the value of minsup was 0.035 and 0.040, the impact on the number of FSITWs was the same.

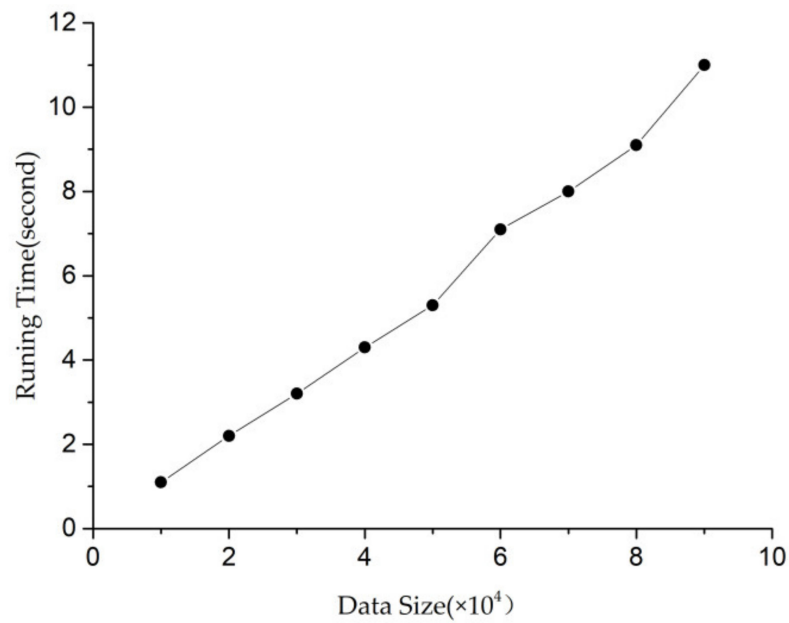


Figure 11. Relationship between “data scale” and “running time”.

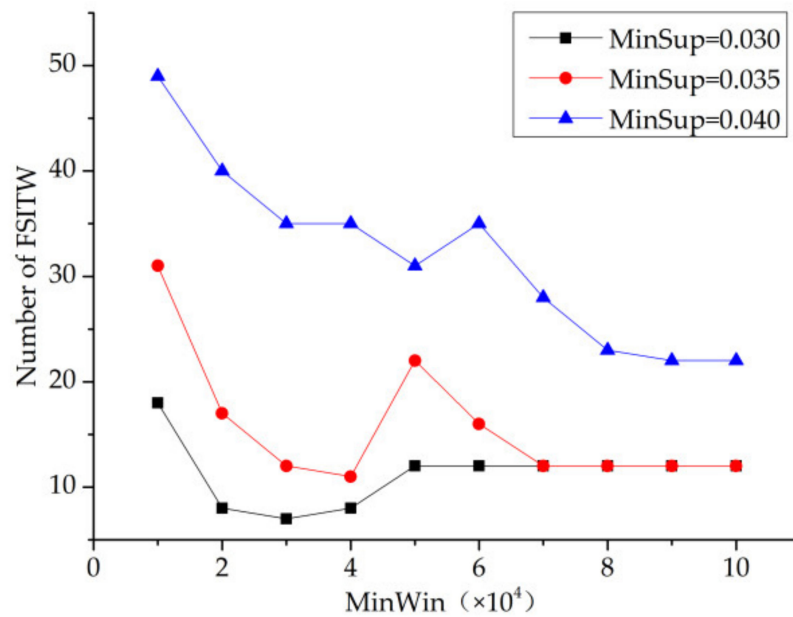


Figure 12. MinWin—number of FSITWs.

Minwin = 100,000 represents that the size of the minimum time window is the total time length of the whole time series database. Moreover, from the curve of minsup = 0.030, we can see that the number of FSITWs finally mined was greater than the number of FSIs.

4.1.3. Analysis of SSPTW Mining Results

In the FSITW mining stage, we found that when minsup = 0.035, the number of FSITWs excavated is the largest. Next, this paper will study the value of minimum confidence because the number of SSPTWs finally mined has a very important relationship with the values of minsup and minconf.

In Figure 13, minsup has three values, which are 0.033, 0.034 and 0.035, respectively. It can be seen from Figure 13 that with the increase in the minsup threshold, the number of SSPTWs finally mined gradually decreases. Moreover, when the minsup threshold is

constant, the number of SSPTWs decreases with the increase in minconf value, but the number of SSPTWs does not change in [0.01,0.4] and [0.06,0.1], which indicates that the confidence has little effect on the number of SSPTWs when it is taken in these two intervals. Finally, this paper selects the representative points of minconf value, which are 0.02, 0.05 and 0.10.

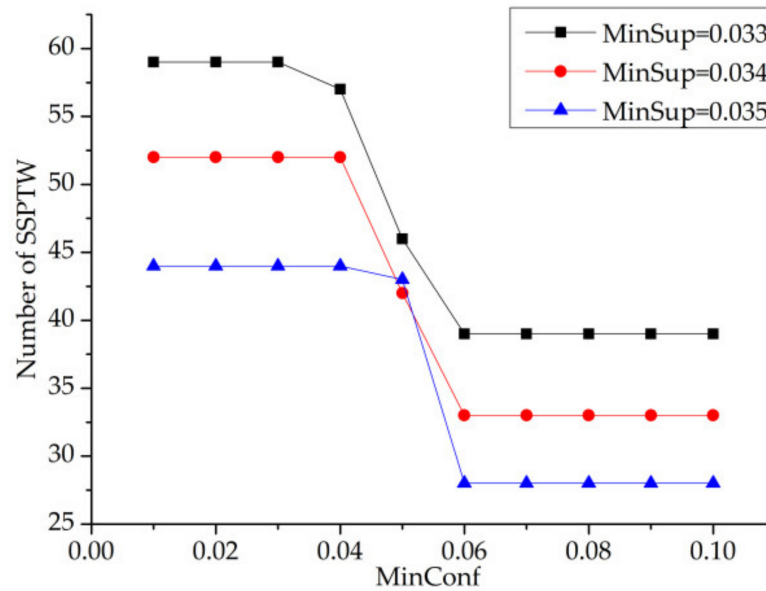


Figure 13. MinConf–MinSup—number of SSPTWs.

In Figures 14 and 15, the values of minconf are 0.02, 0.05 and 0.10, respectively (minwin = 5000 and mintcr = 0.1). From these two figures, we can see that when minsup is constant, the number of SSPTWs decreases with the increase in minconf. However, when minconf is constant, the number of SSPTWs fluctuates greatly. It can be seen from the figure that when minsup = 0.035, the number of SSPTWs finally mined by the three curves is the largest. Therefore, when minsup = 0.035, the number of SSPTWs is greater.

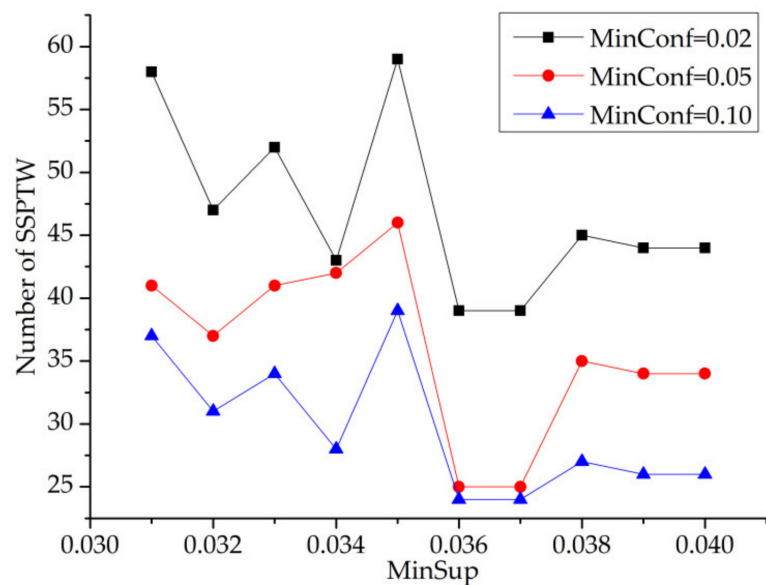


Figure 14. MinSup–MinConf—number of SSPTWs.

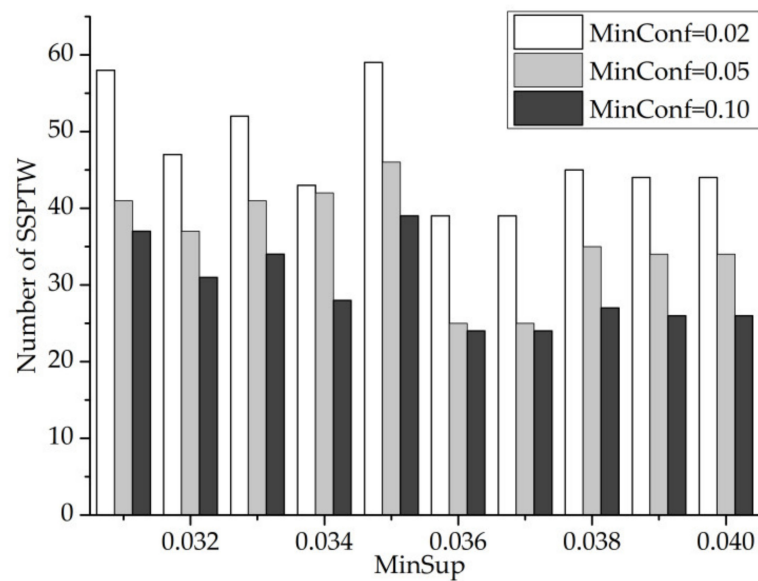


Figure 15. MinSup–MinConf—number of SSPTWs.

4.1.4. Analysis of Results of Strong SSPTW Mining

When the coverage threshold is considered, strong SSPTWs can be mined by this method. In this paper, $minwin = 5000$; $mintcr = 0.1$; and $minsup = 0.33, 0.034$ and 0.035 , respectively.

It can be seen from Figure 16 that, when $minsup$ is constant, the number of strong SSPTWs finally mined decreases with the increase in the $mincov$ value, but when $mincov$ is in the interval of $[0.01, 0.4]$ and $[0.06, 0.1]$, the number curve of strong SSPTWs is a horizontal straight line, which indicates that the change in the $mincov$ value in the interval of $[0.01, 0.04]$ and $[0.06, 0.1]$ has no effect on the number of strong SSPTWs.

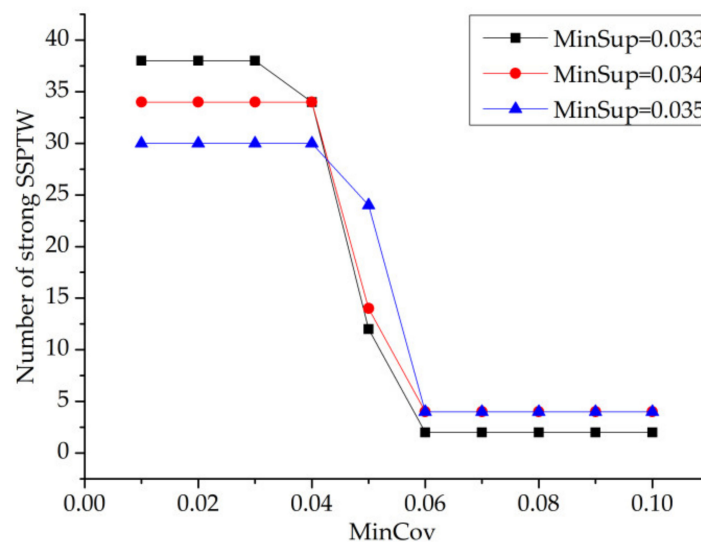


Figure 16. MinCov–MinSup—number of strong SSPTWs.

4.1.5. Analysis of the Impact of the Minimum Time Coverage Threshold on the Number of SSPTWs

This paper studies the value of $mintcr$ and observes the influence of its value on the number of SSPTWs ($minwin = 5000$, $minsup = 0.035$, $minconf = 0.04$).

It can be seen from Figure 17 that the number of SSPTWs decreases with the increase in the $mintcr$ value, but when the $mintcr$ value is in the range of $[0.03, 1.0]$, it has little effect on the number of SSPTWs.

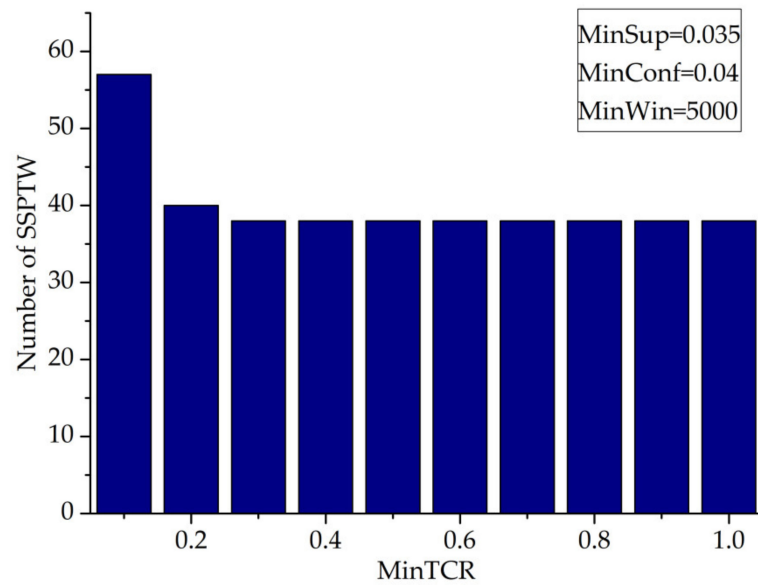


Figure 17. MinTCR—number of SSPTWs.

In Figure 18, we can see the number of SSPTWs in all combinations of mincov–minsup intuitively. In addition, according to the different colors and the depth of colors in the graph, the optimal combination of mincov–minsup is obtained, which is divided into (0.1,0.03) and (0.1,0.035).

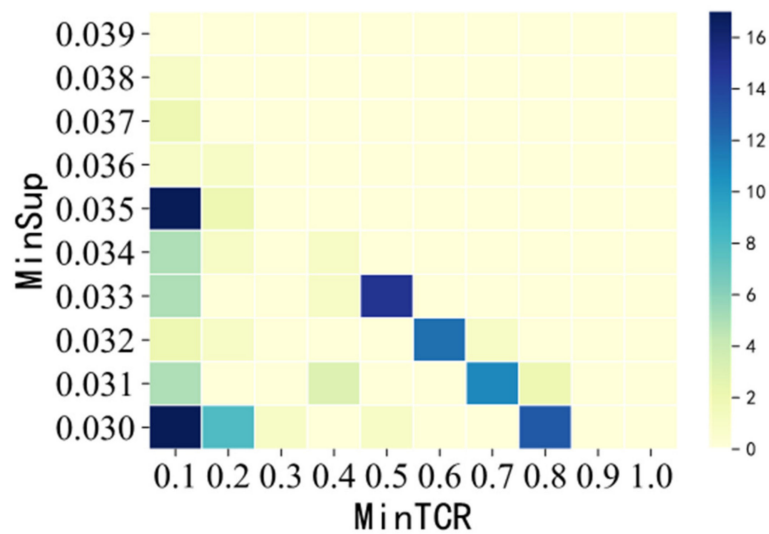


Figure 18. MinTCR–MinSup—number of SSPTWs.

4.1.6. Factor set of FSITW Mining Result Analysis

When considering the ratio constraint value of the factor set, this paper can mine the major factor set of FSITWs (minwin = 5000, minsup = 0.035, minconf = 0.1).

From Figure 19, it can be seen that with the increase in the minfs value, the number of factor sets eventually mined will gradually decrease, and when the minfs value is in three intervals, [0.1,0.3], [0.4,0.6], [0.7,1.0], the change in the corresponding minfs value has no effect on the number of factor sets.

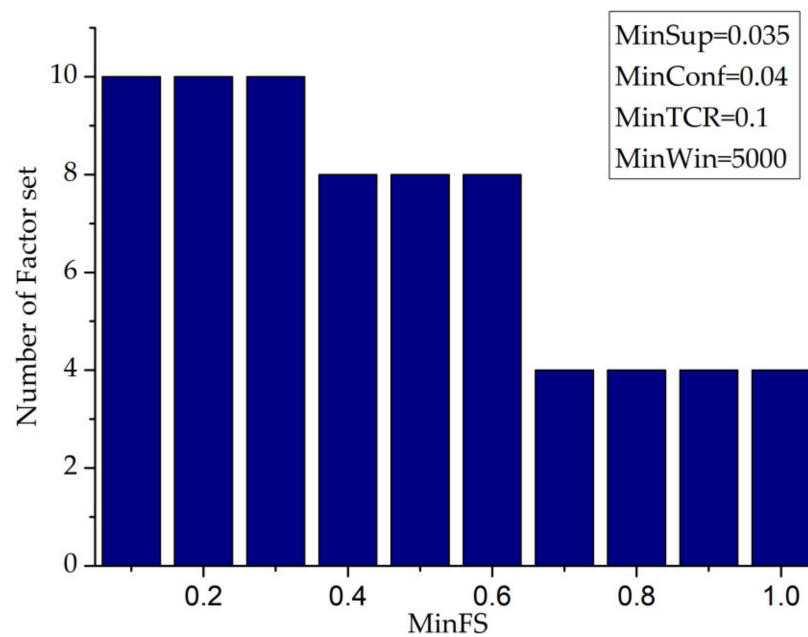


Figure 19. MinFS—number of factor sets.

4.1.7. The Periodicity Analysis of Pattern

By analyzing the periodicity of SSPTWs, we can find the periodic status set sequence patterns which satisfy the cycle width and cycle interval.

Figure 20 shows the number of periodic sequence patterns corresponding to all combinations of “period interval–period width”. Moreover, through the different colors and the changes of color depth, we can intuitively see which combinations ultimately produce the most periodic pattern data so as to provide users with valuable information and facilitate users to make decisions (minwin = 5000, minsup = 0.035, minconf = 0.04, mintcr = 0.1, minptr = 0.1).

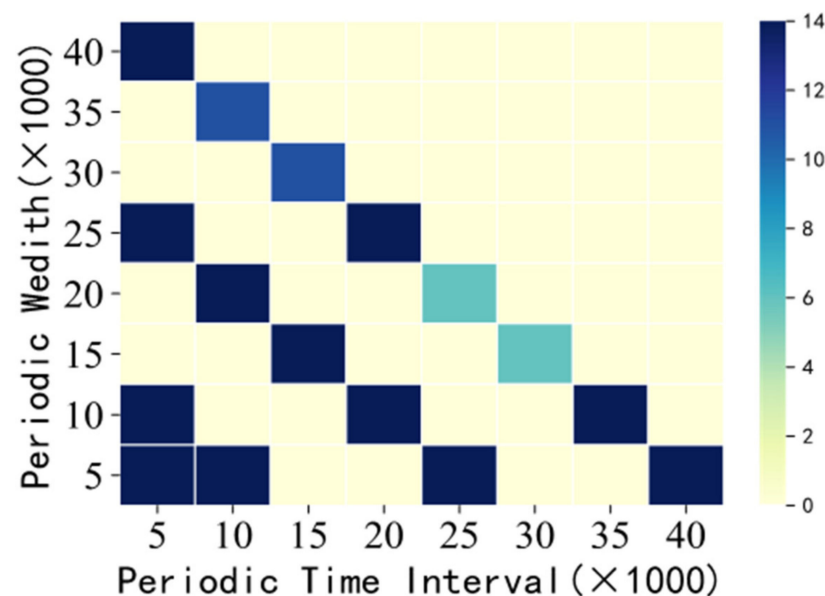


Figure 20. Periodic time interval–periodic width—number of periodic SSPs.

Next, we will fix the value of “cycle width–cycle interval” to study the periodic time coverage and observe the influence of the value of periodic time coverage on the number

of periodic sequence patterns. The cycle width is 5000, and the cycle interval is 5000. From Figure 21, we can see that the number of periodic sequence patterns decreases with the increase in minptcr value. However, when the minptcr value is in the range of [0.1,0.4] and [0.5,0.9], the corresponding minptcr value has little effect on the number of periodic sequence patterns. Additionally, as the minptcr threshold increases, the regularity of the periodic SSPs becomes stronger, and the practical application value becomes greater.

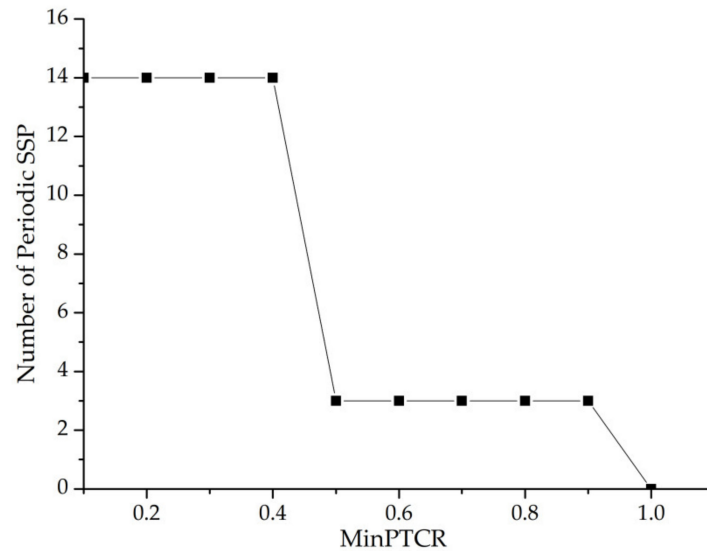


Figure 21. MinPTCR—number of periodic SSPs.

4.2. Comparison and Analysis of Solution Results of SSPM and SSPMTW

4.2.1. Comparative Analysis of FSITW Mining and FSI Mining

Next, this paper will use the TW-Apriori algorithm and Apriori algorithm to mine FSI and make a comparative analysis (minwin = 5000, mintcr = 0.1). The specific figures are shown in Figures 22 and 23.

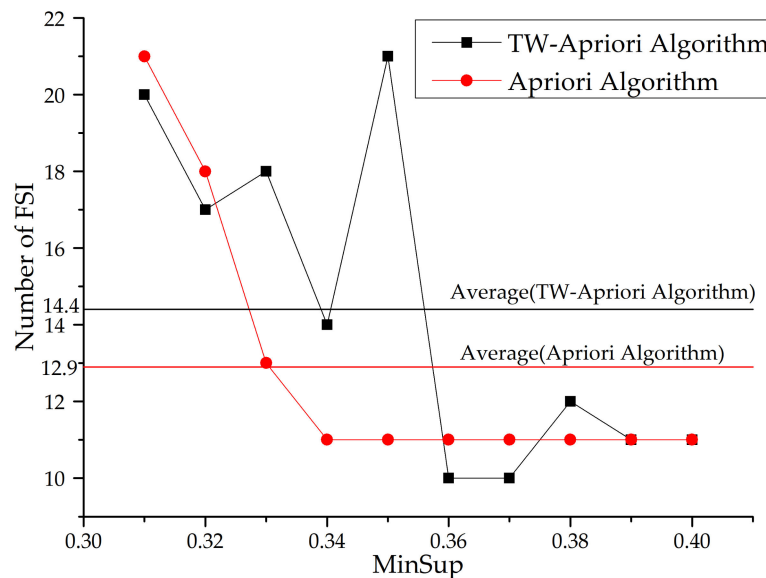


Figure 22. MinSup—number of FSIs.

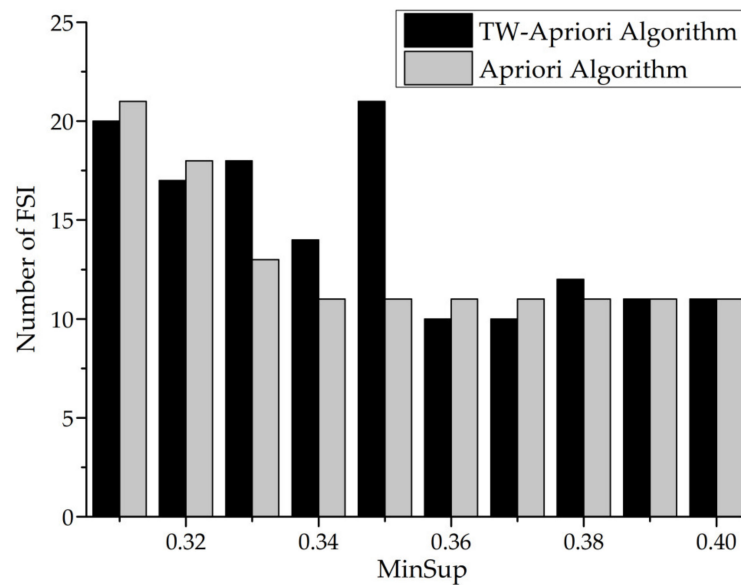


Figure 23. MinSup—number of FSIs.

It can be seen from Figures 22 and 23 that the number of FSI mined by Apriori algorithm decreases with the increase of the minsup threshold, while the mining results of the TW-Apriori algorithm do not show a monotonic decreasing trend, among them; when minsup is greater than 0.35, the number of FSI decreases rapidly; while minsup is greater than 0.37, the number of FSI increases by a small margin. At this time, the number of FSI mined by Apriori algorithm does not change with the growth of minsup. When the value of minsup is in the interval $[0.032, 0.036]$ and $[0.037, 0.039]$, the number of FSIs excavated has three peaks, and in these two intervals, the number of FSIs mined by the TW-Apriori algorithm is more than that mined by traditional algorithms, especially when $\text{minsup} = 0.035$. The TW-Apriori algorithm mined the largest number of FSIs. In addition, it can be seen from Figure 22 that the average value of the mining results of the TW-Apriori algorithm is higher than that of traditional algorithms. In this graph, it is proved that the status set sequential pattern mining method with time window can mine the status itemset in the local time window, which makes up for the defects of traditional sequential pattern mining methods and verifies the important value and significance of SSPMTW.

4.2.2. Comparison and Analysis of SSPTW Mining and SSP Mining

Table 9 shows that the TW-Apriori algorithm and the Apriori algorithm are used to mine status set sequential patterns, respectively. From the last two columns of Table 9, it can be seen that the number of SSPs mined by the two methods shows a downward trend with the increase in the support threshold. When the support threshold is 3.0%, the number of SSPs mined by the two methods is the largest, and the number of SSPs mined by SSPMTW is 40. The number of SSPs mined by traditional methods is 31; when the support threshold is the same, the number of SSPs with time windows is more than that without time windows. The average number of SSPs with time windows is 12.7, while the average number of SSPs without time windows is 5.1. The red words in the table indicate that when the minsup value is in the interval $[3.4\%, 3.9\%]$ and the mintcr value is in the interval $[10\%, 30\%]$ the traditional method was unable to mine SSPs, while SSPMTW can still mine a certain number of SSPs, which means that the traditional method ignores the sequence patterns existing in the local time window, and the SSPMTW proposed in this paper can solve this problem well.

Table 9. Comparative analysis of the two algorithms under the constraint of “mintcr–minsup”.

Number of SSPs	MinTCR (TW-Apriori)									Total	Apriori
	10%~20%	20%~30%	30%~40%	40%~50%	50%~60%	60%~70%	70%~80%	80%~90%	90%~100%		
3.0%	17	8	1	0	1	0	0	13	0	40	31
3.1%	4	0	0	3	0	0	11	2	0	20	12
3.2%	2	1	0	0	0	12	1	0	0	16	6
3.3%	4	0	0	1	16	0	0	0	0	21	2
3.4%	4	1	0	1	0	0	0	0	0	6	0
3.5%	17	2	0	0	0	0	0	0	0	19	0
3.6%	1	1	0	0	0	0	0	0	0	2	0
3.7%	2	0	0	0	0	0	0	0	0	2	0
3.8%	1	0	0	0	0	0	0	0	0	1	0
3.9%	0	0	0	0	0	0	0	0	0	0	0

Therefore, through comparative analysis, it was found that because SSPM is carried out over the whole time period of the time series database, when the support threshold increases, it may not be able to mine SSPs that meet the constraints; however, SSPMTW is carried out over the local time window, so it can mine some SSPs that do not meet the constraints in the whole time period but do meet the constraints in the local time window.

4.2.3. Analysis of the Influence of Time Window and Non-Time Window on Pattern Diversity

SSPM without considering time windows can mine FSIs, SSPs, strong SSPs and factor sets of FSIs, while SSPMTW can not only mine FSITWs, SSPTWs, strong SSPTWs and factor sets of FSITWs but can also mine periodic SSPs.

Therefore, compared with SSPM, SSPMTW can mine the rules existing in the local time window and analyze the periodicity of the rules.

5. Discussion and Conclusions

5.1. Discussion

Experiments show that the proposed method can discover the relationship between items with different attributes. The algorithm proposed in this paper can mine more frequent itemsets and sequential patterns than the traditional algorithm, which shows that the traditional sequential pattern mining methods ignore the rules in the local time window, and the method proposed in this paper makes up for this. In addition, by adding the confidence threshold, coverage threshold and factor set ratio, and analyzing the periodicity of the status set sequence pattern with time windows, this paper has revealed the strong status set sequence pattern with time windows, the main factor set of frequent status itemsets with time windows and the periodic status set sequence patterns; these rules are more regular and valuable, overcoming the limitations of traditional sequential pattern mining algorithms. These more regular sequential patterns can better predict the future state of the system, make the operation of the system more stable and orderly, and reduce the entropy of the system.

The mining problem of the rule of “heatstroke→complication” mentioned above can be solved well by the method proposed in this paper. The occurrence of heatstroke and its complications is not frequent over the whole period of time. After dividing the whole study period into months, we found that the occurrence of these diseases was frequent in the months with higher temperatures. On this basis, the corresponding laws can be easily mined out. According to these rules, appropriate preventive measures can be taken for these diseases. Therefore, the method and algorithm proposed in this paper can solve some practical problems well and have certain practical significance.

5.2. Conclusions

In this paper, the problem of mining sequential patterns of status sets with time windows was proposed by considering the case of time windows. Firstly, the concepts involved in SSPMTW process were defined, and the corresponding properties were proposed, including some new constraints, such as minimum time window, minimum time coverage and minimum periodic time coverage. The periodicity of SSPMTWs was analyzed, and the constraints of cycle width and interval are proposed. According to the research content, this paper proposes the TW-Apriori algorithm and explains the idea of the algorithm. Finally, through small-scale and large-scale examples, the feasibility, effectiveness and efficiency of the proposed method and algorithm were verified, and a variety of patterns and rules were finally mined, such as FSITW, SSPTW, strong SSPTW, factor set of FSITW and periodic SSP; these rules can make the system more orderly and reduce the entropy of the system. Through the comparative analysis of SSPM and SSPMTW, it can be seen that compared with SSPM, SSPMTW can excavate the laws existing in the local time window and analyze the periodicity of the laws, which solves the problem of SSPM ignoring the laws existing in the local time window. In addition, by increasing the confidence threshold, coverage threshold and factor set ratio, we discovered more regular and valuable sequential patterns in the local time window, which overcomes the limitations of traditional sequential pattern mining methods.

Author Contributions: Conceptualization, H.L.; methodology, H.L.; software, H.L. and W.H.; validation, B.C., Y.X. and Y.Z.; formal analysis, H.L., X.J. and B.C.; investigation, W.C.; resources, Y.X.; data curation, Y.X.; writing—original draft, H.L.; writing—review and editing, Y.X.; visualization, H.L. and W.H.; supervision, S.Z. and W.H.; project administration, S.Z.; funding acquisition, S.Z. All authors have read and agreed to the published version of the manuscript.

Funding: This research was funded by the National Natural Science Foundation of China (Grant No. 71971013 and 71871003) and the Fundamental Research Funds for the Central Universities (YWF-20-BJ-J-943). This research was also funded by the Graduate Student Education & Development Foundation of Beihang University.

Institutional Review Board Statement: Not applicable.

Informed Consent Statement: Not applicable.

Data Availability Statement: Not applicable.

Conflicts of Interest: The authors declare no conflict of interest.

References

1. Xiao, Y.; Zhang, R.; Kaku, I. A new framework of mining association rules with time-window on real-time transaction database. *Int. J. Innov. Comput. Inf. Control.* **2011**, *7*, 3239–3253.
2. Xiao, Y.; Tian, Y.; Zhao, Q. Discovering Frequent Itemset with Maximum Time-Window on Temporal Transaction Database using Variable Neighborhood Search. *Electron. Notes Discret. Math.* **2012**, *39*, 137–144. [[CrossRef](#)]
3. Xiao, Y.; Tian, Y.; Zhao, Q. Optimizing frequent time-window selection for association rules mining in a temporal database using a variable neighbourhood search. *Comput. Oper. Res.* **2014**, *52*, 241–250. [[CrossRef](#)]
4. Zabihi, F.; Ramezan, M.; Pedram, M.M.; Memariani, A. Fuzzy sequential pattern mining with sliding window constraint. In Proceedings of the Institute of Electrical and Electronics Engineers (IEEE), Shanghai, China, 22–24 June 2010; Volume 5, pp. 5–396.
5. Ozden, B.; Ramaswamy, S.; Silberschatz, A. Cyclic association rules. In Proceedings of the 14th International Conference on IEEE, Orlando, FL, USA, 23–27 February 1998; pp. 412–421.
6. Li, D.; Deogun, J.S. Discovering Partial Periodic Sequential Association Rules with Time Lag in Multiple Sequences for Prediction. *Comput. Vision* **2005**, *3488*, 332–334.
7. Nishi, M.A.; Ahmed, C.F.; Samiullah, M.; Jeong, B.-S. Effective periodic pattern mining in time series databases. *Expert Syst. Appl.* **2013**, *40*, 3015–3027. [[CrossRef](#)]
8. Han, J.; Dong, G.; Yin, Y. Efficient mining of partial periodic patterns in time series database. In Proceedings of the 15th International Conference on IEEE, Sydney, Australia, 23 March 1999; pp. 106–115.
9. Li, Y.; Ning, P.; Wang, X.S.; Jajodia, S. Discovering calendar-based temporal association rules. *Data Knowl. Eng.* **2003**, *44*, 193–218. [[CrossRef](#)]

10. Yang, J.; Wang, W.; Yu, P. Mining asynchronous periodic patterns in time series data. *IEEE Trans. Knowl. Data Eng.* **2003**, *15*, 613–662. [[CrossRef](#)]
11. Huang, K.-Y.; Chang, C.-H. SMCA: A general model for mining asynchronous periodic patterns in temporal databases. *IEEE Trans. Knowl. Data Eng.* **2005**, *17*, 774–785. [[CrossRef](#)]
12. Lee, W.-J.; Jiang, J.-Y.; Lee, S.-J. Mining fuzzy periodic association rules. *Data Knowl. Eng.* **2008**, *65*, 442–462. [[CrossRef](#)]
13. Seno, M.; Karypis, G. Finding frequent patterns using length-decreasing support constraints. *Data Min. Knowl. Discov.* **2005**, *10*, 197–228. [[CrossRef](#)]
14. Wang, J.; Karypis, G. Bamboo: Accelerating closed itemset mining by deeply pushing the length-decreasing support constraint. In Proceedings of the 2004 SIAM International Conference on Data Mining, Orlando, FL, USA, 22 March 2004.
15. Yun, U.; Leggett, J.J. WLP Miner: Weighted frequent pattern mining with length-decreasing support constraints. *Adv. Knowl. Discov. Data Min.* **2005**, *3518*, 555–567.
16. Yun, U. WIS: Weighted interesting sequential pattern mining with a similar level of support and/or weight. *ETRI J.* **2007**, *29*, 336–352. [[CrossRef](#)]
17. Yun, U. An efficient mining of weighted frequent patterns with length decreasing support constraints. *Knowl. Based Syst.* **2008**, *21*, 741–752. [[CrossRef](#)]
18. Chang, J.H. Mining weighted sequential patterns in a sequence database with a time-interval weight. *Knowl. Based Syst.* **2011**, *24*, 1–9. [[CrossRef](#)]
19. Massegli, F.; Poncelet, P.; Teisseire, M. Incremental mining of sequential patterns in large databases. *Data Knowl. Eng.* **2003**, *46*, 97–121. [[CrossRef](#)]
20. Hand, D.; Ng, W.K.; Kitsuregawa, M.; Jianzhong, L.; Chang, K. Sequential Pattern Mining with Time Intervals. In *Pacific Asia Conference on Knowledge Discovery and Data Mining*; Springer: Berlin/Heidelberg, Germany, 2006; pp. 775–779.
21. Zhang, J.; Wang, Y.; Yang, D. CCSpan: Mining closed contiguous sequential patterns. *Knowl. Based Syst.* **2015**, *89*, 1–13. [[CrossRef](#)]
22. Fabregue, M.; Braud, A.; Bringay, S.; Le Ber, F.; Teisseire, M. Mining closed partially ordered patterns, a new optimized algorithm. *Knowl. Based Syst.* **2015**, *79*, 68–79. [[CrossRef](#)]
23. Agrawal, R.; Srikant, R. Fast Algorithms for Mining Association Rules. In Proceedings of the 20th International Conference on Very Large Data Bases, Santiago, Chile, 12–15 September 1994; pp. 487–499.
24. Srikant, R.; Agrawal, R. Mining Sequential Patterns: Generalizations and Performance Improvements. In Proceedings of the 5th International Conference on Extending Database Technology, Avignon, France, 25–29 March 1996; pp. 3–17.
25. Massegli, F.; Cathala, F.; Poncelet, P. The PSP Approach for Mining Sequential Patterns. In Proceedings of the 2nd European Symposium on Principles of Data Mining and Knowledge Discovery, Nantes, France, 23–26 September 1998; pp. 176–184.
26. Zaki, M. SPADE: An Efficient Algorithm for Mining Frequent Sequences. *Mach. Learn.* **2001**, *42*, 31–60. [[CrossRef](#)]
27. Ayres, J.; Flannick, J.; Gehrke, J.; You, T. Sequential Pattern Mining Using a Bitmap Representation. In Proceedings of the 8th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, Edmonton, AB, Canada, 23–26 July 2002; pp. 429–435.
28. Yu, H.; Wen, J.; Wang, H.; Jun, L. An Improved Apriori Algorithm Based on the Boolean Matrix and Hadoop. *Procedia Eng.* **2011**, *15*, 1827–1831. [[CrossRef](#)]
29. Djenouri, Y.; Comuzzi, M. Combining Apriori heuristic and bio-inspired algorithms for solving the frequent itemsets mining problem. *Inf. Sci.* **2017**, *420*, 1–15. [[CrossRef](#)]
30. Du, Z. Energy analysis of Internet of things data mining algorithm for smart green communication networks. *Comput. Commun.* **2020**, *152*, 223–231. [[CrossRef](#)]
31. Vasoya, A.; Koli, N. Mining of Association Rules on Large Database Using Distributed and Parallel Computing. *Procedia Comput. Sci.* **2016**, *79*, 221–230. [[CrossRef](#)]
32. Guevara-Cogorno, A.; Flamand, C.; Alatrasta-Salas, H. Copper—Constraint Optimized Prefixspan for Epidemiological Research. *Procedia Comput. Sci.* **2015**, *63*, 433–438. [[CrossRef](#)]