Advances in Computer, Signals and Systems (2025)
Clausius Scientific Press, Canada

DOI: 10.23977/acss.2025.090319
ISSN 2371-8838 Vol. 9 Num. 3

A Review of Research on Pruning Strategies in Maximum Frequent Itemset Mining Algorithms

Fentian Li

The Tourism College of Changchun University, Changchun, Jilin, 130607, China
2812001856@qq.com

Keywords: Frequent Itemsets; Enumeration Tree; Pruning Strategy; Maximum Frequent Itemsets

Abstract: The mining of frequent itemsets has become a hot topic among researchers. If the process of mining frequent itemsets is regarded as a search problem, then the search space is an enumeration tree. In order to minimize unnecessary nodes for search, the optimization of pruning technology can improve the mining efficiency of frequent itemsets to a certain extent. This is one of the important means to improve the efficiency of frequent itemsets. This article improves the definitions of frequent itemsets and enumeration trees, analyzes the use of various pruning strategies, and summarizes the efficiency of various pruning strategies for mining maximum frequent itemsets.

1. Introduction

With the development of the big data era, the role of data mining technology in this field cannot be underestimated. Discovering association rules is the focus of data mining work. The mining of frequent patterns is the core and foundation of association mining [1]. It is a decisive factor affecting the efficiency of mining algorithms. It is the basis for generating association rules. Therefore, any progress in frequent pattern mining will have an important impact on the efficiency of association mining and other data mining tasks. However, under normal circumstances, the number of frequent item sets obtained is huge, and items are repeated among each other, which leads to the problem of information redundancy. Since the maximum frequent itemset contains all frequent itemsets, which greatly reduces the number of frequent itemsets, the problem of calculating frequent itemsets can be transformed into calculating the maximum frequent itemsets. In some applications, only the maximum frequent itemset is needed and not all frequent itemsets are needed. In this way, it is of great significance to study the algorithm for directly calculating the maximum frequent itemsets [2].

How to effectively prune the search space is a core of research work on maximum frequent itemset mining. Choosing an appropriate pruning strategy can effectively reduce the search space during the mining process. This paper reviews the pruning strategies of maximum frequent itemsets from three aspects: using non-frequent subset pruning, using frequent superset pruning, and using parent-child relationship pruning. Clever selection of pruning strategies during the mining process can improve the efficiency of the algorithm [3].

2. Basic concepts and theories

2.1 Concept explanation

Definition 1 Assume that I={I1,I2,I3,...,Im} is a set of all items, where each item Ii ($1 \le i \le m$) in I is called an item. If the set $X \subseteq I$, then X is called an itemset.

Definition 2 Definition 2 Given an item set $X \subseteq I$, the number of times X appears in T is taken as the support of the item set

Definition 3 Let the minimum support threshold be min_sup. min_sup can be either a specific value or expressed as a percentage. Assuming that the number of transactions in the database is M and min_sup is expressed as a percentage, the minimum support threshold can be obtained by multiplying the minimum support percentage by the number of transactions M. If min_sup=2%, and there are 100 transactions in total, then when the number of occurrences of an item set is greater than or equal to 2, we consider it to be frequent. Of course, we can also set min_sup=2 directly. In this article, we set min_sup to a specific value. If $Sup(X) \ge min\_sup$, then X is said to be a frequent itemset FI [4].

Definition 4: Each item of set X is included in set Y, but at least one item in set Y is not in set X. Y is called a true superset of X, and X is a true subset of Y.

Definition 5 If there is no true superset Y such that sup(Y)=sup(X), and the itemset X is frequent, then the itemset X is said to be a closed frequent itemset CFI in the database.

Definition 6 If the itemset X is frequent in the database, and for any itemset Y that satisfies $X \subset Y$, Y is a superset of X and is infrequent, then X is the maximum frequent itemset MFI.

2.2 Theoretical properties

Definition 7 Enumeration tree refers to arranging all items appearing in the database in dictionary order and logically organizing them into the form of an enumeration tree. If item i appears before item j in set I, then i < j. "<" is used to indicate the lexicographic order on the item set I. Level 0 is the root and is empty. The kth layer contains all k itemsets. Each node in the tree is composed of two parts, namely the head of the node and the tail of the node. Head is actually the item set represented by the node itself. Tail is a collection of items that can be expanded by a node. Note that Tail contains all item elements that are lexicographically greater than Head. As shown in Figure 1, assume that the set of all items is {a, b, c}, the "()" in the figure represents Head, and the "{}" represents Tail.

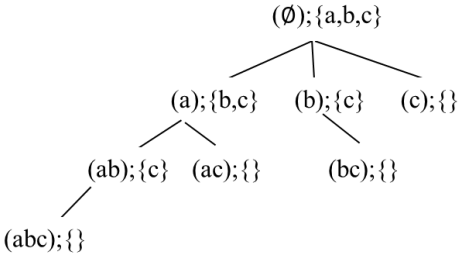

Figure 1: enumeration tree

As shown in Figure 1, it is a set enumeration tree of {a, b, c}. In this enumeration tree, it is assumed that these three elements satisfy the relationship of a, b, c increasing in order. The unique ordered relationship of items represents the parent-child relationship of nodes in the enumeration tree. In the figure above, what we assume is a relatively static sequential correlation. In a specific algorithm, if we want to improve the efficiency of the algorithm, we can do this by dynamically adjusting the corresponding relationship. Obviously, if the order in which the elements in the set are defined is

different, the corresponding set enumeration tree will also be different [5].

Property 1 (Apriori property) All non-empty subsets of frequent itemsets must also be frequent.

Property 1 has anti-monotonicity, that is, if X is a non-frequent itemset, then any superset Y of X is a non-frequent itemset. During the deep search process, the enumeration tree can be pruned using Property 1 and its anti-monotonicity.

Property 2: If an itemset is not a maximum frequent itemset, then it is not completely certain whether its subset can be a maximum frequent itemset, but the subset has the possibility of being a frequent itemset, and of course it may also be an infrequent itemset. Therefore, it is necessary to determine the maximum frequent itemset of the subset in turn.

Property 3: Let the current node be C, $i \in Tail(C)$. If $Sup(C \cup \{i\})=Sup(C)$ and $Sup(C \cup \{i\}) \geq min\_sup$, you can directly point the current node to $C \cup \{i\}$. This property can be used to prune the parent-child equivalence relationship.

3. Pruning strategy

The mining of maximum frequent itemsets can actually be regarded as a search problem, and its search space is an enumeration tree. According to the definition of enumeration tree, if there are m items in the database, then the enumeration tree will have 2m nodes to be searched. In order to reduce unnecessary search nodes as much as possible, pruning optimization technology can improve the exploration efficiency of frequent item sets, which is an important means to improve the efficiency of frequent item set mining. For the huge search space that exists, pruning becomes an advantageous method for mining frequent itemsets [6].

The enumeration tree is a hierarchical tree structure, combined with the definition of the enumeration tree. Each node is expanded by the nodes of the previous layer in a certain order. This order is called lexicographic order. The meaning represented by the Head of a node is the item set of each node. The Tail of a node refers to the possible lexicographically extended set of each node, $Tail=\{x|x \in I \text{ and } y \in Head \text{ and } y<x\}$, that is, the elements contained in the Tail are greater than each element in the Head in lexicographic order. The candidate extended item set $CE(N)=\{x|x \in I \text{ and } x \notin Head \text{ and } N \cup \{x\} \text{ of node N may be frequent}\}$, and $CE(N)=Tail(N)$ can be obtained from analysis. The frequent extended itemset $FE(N)=\{x|x \in CE(N) \text{ and } N \cup \{x\} \text{ of node N is frequent}\}$ and is represented by "[]" in the figure.

3.1 Pruning using infrequent subsets

3.1.1 Basic pruning strategy 1 (BasicPS1)

For the current node N, through support calculation, the frequent expansion branches of N are retained and the infrequent expansion branches of N are deleted. The specific operation of support calculation is as follows: according to the dictionary order, the support of $N \cup \{x\}(x \in CE(N))$ is calculated sequentially. If $sup(N \cup \{x\}) \geq min\_sup$, retain the $N \cup \{x\}$ branch; otherwise, delete this branch. In other words, the child nodes of N are composed of $N \cup \{x\}$, where $x \in FE(N)$.

Since it is meaningless to perform any operation on infrequent itemsets. Therefore, pruning infrequent expansion branches will avoid many unnecessary expansions and can effectively reduce the size of the search space without losing the information of frequent itemsets. As shown in Figure 2, $I=\{a,b,c,d\}$, node $N=\{a\}$, then $CE(N)=\{b,c,d\}$. Through support calculation, it is known that {ab} and {ad} are frequent item sets, while {ac} is a non-frequent item set. Based on the basic pruning strategy 1, there is no need to continue to expand {ac} and can be deleted. At the same time, the candidate extension set $FE(N)=\{b,d\}$ of node N is obtained.

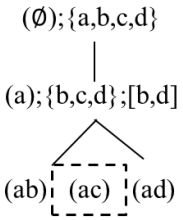

Figure 2: BasicPS1 (the dotted box is the pruned node)

To sum up, this strategy is essentially a process of obtaining FE(N), which can be applied to most frequent itemset mining algorithms. It has universal applicability and can improve the efficiency of the algorithm on the basis of reducing the search space. It is especially suitable for mining algorithms using the depth-first strategy. However, the shortcoming of this strategy is that when traversing node N, FE(N) must be obtained from CE(N) through support calculation. Pruning is conditional, and the condition is that it can only be realized when there is non-frequent expansion. In the above figure2, only after the itemset {ac} is supplemented and the support is calculated, it is discovered that the set is a non-frequent itemset. This approach produces a large number of non-frequent itemsets that have no effect [7].

3.1.2 Basic pruning strategy 2 (BasicPS2)

In the enumeration tree, let N be the current node and its parent node is P, then $N=P \cup \{x\}$, $x \in FE(P)$. CE(N) comes from FE(P) of its parent node P. That is, $CE(N)=\{y|y \in FE(P) and x<y\}$.

As shown in Figure 3, node N={a}, according to the definition of lexicographic subset enumeration tree, CE(N)={b,c,d}, that is to say, {ab}, {ac} and {ad} should be extended at the next level of N. However, N's parent node P={∅}, it is known through support calculation that FE(P)={a,b,d}, and {c} is infrequent. According to the Apriori characteristics, it can be inferred that {ac} is also infrequent. Therefore, according to the basic pruning strategy 2, CE (N) = {b, d}, the number of items in the candidate extended item set is reduced, so that the extended item set {ac} will not be obtained, thereby avoiding repeated calculation of support for infrequent extended item sets, effectively reducing the amount of calculation, and further improving the execution speed of the algorithm.

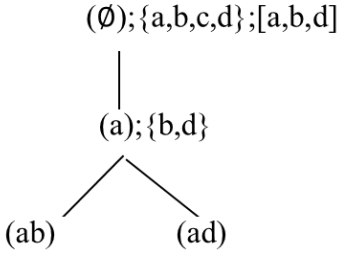

Figure 3: BasicPS2

Comparing the two basic pruning strategies, we can see that BasicPS2 can reduce the candidate extension set CE of its own node based on the frequent extension set FE of the parent node. Basic pruning strategy 2 can be regarded as an effective supplement to basic pruning strategy 1. Basic pruning strategy 2 is established after basic pruning strategy 1 generates the frequent expansion set FE of the parent node. There is a logical progressive relationship between the two pruning strategies. They generally appear at the same time in the algorithm for mining frequent item sets. The cross-use of each other can greatly reduce the search space. However, in each candidate candidate extended item set, only one item is taken for support calculation operation, which makes the algorithm face a huge amount of calculation, which is an inevitable problem of the basic pruning strategy.

In the context of the digital economy, the continuous emergence of new technologies and new business models has caused changes on the demand side, and customer demand is also shifting towards intelligence and digitalization. To a certain extent, this forces enterprises to use digitalization to empower themselves and carry out digital transformation and upgrading. Changes in consumer demand have forced the equipment manufacturing industry to transform and upgrade. Driven by changes in demand, the equipment manufacturing industry uses digital transformation to empower itself is an inevitable choice for it to meet customer needs and improve its comprehensive competitive strength [8].

3.2 Exploiting frequent superset pruning

The theoretical basis of this type of pruning strategy is that the subsets of property 1 frequent itemsets are also frequent. For the maximum frequent itemset, once it is determined to be the maximum frequent itemset, there is no need to obtain its subsets.

3.2.1 Maximum pruning strategy 1 (MaxPS1)

For the current node N, let $M=N \cup CE(N)$. If $sup(M) \geq min\_sup$, then M is frequent. Then the subtree with N as the root node does not need to be traversed and is suitable for mining the maximum frequent itemset.

This strategy is based on the definition of maximum frequent itemsets and the Apriori property. In the enumeration tree, if M is frequent, then its subsets are also frequent. At the same time, in the subtree rooted at N, there will be no itemset larger than M. Therefore, in the process of mining the maximum frequent itemsets, only the maximum of M can be detected. As shown in Figure 4, for node $N=\{a\}$, before extending it, first calculate the support of $N \cup CE(N)=\{abcde\}$. It is assumed here that $sup(\{abcde\}) \geq min\_sup$, then $N \cup CE(N)$ is frequent. According to the maximum pruning strategy 1, node {a} does not need to continue to expand, and the subtree can be deleted from the search space.

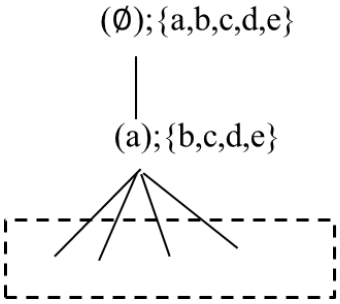

Figure 4: MaxPS (The dotted box is the extended branch that can be pruned)1

This strategy can effectively reduce the search space in the maximum frequent itemset mining algorithm, but it requires scanning the database when making support judgments, which increases I/O operations. Moreover, it can only be tentatively designated as the candidate maximum frequent itemset, and it cannot be determined whether it is the maximum frequent itemset. This still needs to be judged through global operations [9].

3.2.2 Maximum pruning strategy 2 (MaxPS2)

In the enumeration tree, the current node is N. If $M=N \cup CE(N)$ is a subset of a certain maximum frequent itemset, then M must be frequent. Then the subtree rooted at N can be subtracted. Because starting from node N, the expansion obtained are all subsets of M, and there will be no frequent itemsets larger than M. Therefore, this method is conducive to and can more effectively mine the

maximum frequent itemset.

This method is based on two theories: the concept of maximum frequent itemsets and the Apriori property. First, a part of the maximum frequent itemset will be obtained through the traditional method, as shown in Figure 5. Assume {abcd} is a maximum frequent itemset, then when expanding the N={b} node, it is found that $N \cup CE(N)=\{bcd\} \subseteq \{abcd\}$, then delete the subtree with the {b} node as the root. In the same way, for nodes {c}, {d}, the union of them and their respective candidate extension sets is also included in {abcd}, so it can also be pruned. The final result is actually a path from the root node to {abcd}.

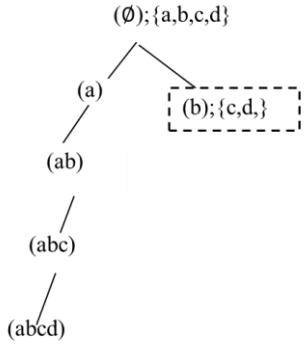

Figure 5: MaxPS2 (The dotted box is the pruned node)

Maximum pruning strategy 2 can greatly reduce the search space during the maximum frequent itemset search process. Because this process does not need to calculate support, it greatly reduces the number of database scans and improves efficiency. Especially when the database size is large, the execution speed of the algorithm will be significantly improved. But the disadvantage is that detection methods must be used. Every time a node is expanded or supplemented, it is necessary to check whether the union of the node and its candidate expansion set forms a subset of a known maximum frequent item set. In the process of increasing the maximum frequent itemset, the detection space continues to expand, which increases the algorithm overhead. Because this pruning strategy has a predictive nature, if the maximum frequent itemset is a node in the deepest layer, each prediction step in the previous operation is a waste of time. Therefore, this method can be integrated into this pruning strategy after obtaining a certain amount of maximum frequent itemsets.

From the comparison, it can be seen that both MaxPS1 and MaxPS2 operate on the "merging" of the current node and its candidate extension set. MaxPS2 can also be regarded as a supplementary operation of MaxPS1. The candidate maximum frequent itemset is first obtained through MaxPS1. However, MaxPS2 does not perform support calculation, while MaxPS1 does. Therefore, MaxPS1 is suitable for small databases, and MaxPS2 is suitable for large databases [10].

3.2.3 Maximum pruning strategy 3 (MaxPS3)

This strategy is also called the lookahead pruning strategy. If $N \cup CE(N)$ is frequent, all sibling nodes of N can be deleted.

Assume that node M is an adjacent node of node N, and node P is the parent node of nodes N and M, then $N=P \cup ij$, $N=P \cup ik$, $ij, ik \in CE(P)$, and $CE(M) \in CE(N) \in CE(P)$; thus we get: $M \cup CE(M)=P \cup \{ik\} \cup CE(M) \subseteq P \cup CE(N) P \cup \{ij\} \cup CE(N)=N \cup CE(N)$, that is, $M \cup CE(M)$ is a proper subset of $N \cup CE(N)$. Since $N \cup CE(N)$ is frequent, according to the characteristics of frequent itemsets, $M \cup CE(M)$ is frequent. And $|N \cup CE(N)|>|M \cup CE(M)|$, obviously, $M \cup CE(M)$ is a frequent itemset but not a maximum frequent itemset, so no new maximum frequent itemsets will appear when searching for node M and its corresponding expansion space. Therefore, if $N \cup CE(N)$ is frequent, the adjacent

nodes on the right side of node N (such as node M) and its extended subtree can be pruned, as shown in Figure 6.

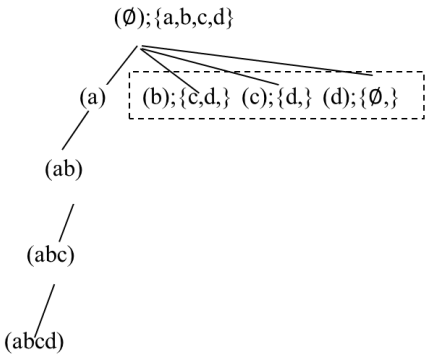

Figure 6: MaxPS3 (The dotted box is the pruned node)

This strategy is a progressive pruning strategy of MaxPs1 and MaxPs2. Different from the first two, MaxPs3 can not only delete its child nodes, but also delete its sibling nodes at the same level. After theoretical proof, the same judgment criteria can delete more useless branches. The method of determining its frequency is also flexible, which can be determined by calculating the support or by determining whether it is a subset of the existing maximum frequent itemset.

MaxPs3 is an effective extension of MaxPs1 and MaxPs2. MaxPs1 and MaxPs2 mainly use the prediction maximization strategy when node N is expanded downward to judge the frequency of N $\cup$ CE(N) in advance. If it is frequent, N will not be expanded in any way. On this basis, 3 can not only cut off the expansion space of the N node, but also further cut off the expansion space of the adjacent nodes on the right side of N. In some algorithms, the combined use of MaxPs1, MaxPs 2 and MaxPs 3 can greatly improve the efficiency of pruning [11].

3.2.4 Depth-first maximum pruning strategy (DFMaxPS)

In the depth-first algorithm, P is the parent node of N N=P $\cup$ {x}, x$\in$FE(P). When N nodes and their extended subtrees are traversed, the support of P $\cup$ {y|y$\in$FX(P) and x<y} is calculated. If it is frequent, in the subtree with P as the head node, all nodes on the right side of N and their extended branches can be pruned. At this time, you only need to check whether P $\cup$ {y|y$\in$FX(P)andx<y} is the maximum frequent itemset. Assume that depth-first search always starts from the left subtree. Being adjacent to N means that it is at the same level as N and has the same parent node as N, that is, it is expanded and generated by the same parent node.

The depth-first maximum pruning strategy is very similar to the maximum pruning strategy 3. They both prune the search space expanded by all "adjacent" nodes to the right of node N. The difference is that maximum pruning strategy 3 prunes the adjacent nodes on the right side of node N and its extended branches by judging whether the item set N $\cup$ CE(N) is a frequent item set. However, if N $\cup$ CE(N) is not a frequent item set, this strategy cannot prune the adjacent nodes on the right side of node N and its extended branches. It can only be processed by relying on maximum pruning strategies 1 and 2. The depth-first maximum pruning strategy uses the parent node P of node N to prune the branch on the right side of node N by calculating whether P $\cup$ {y|y$\in$FX(P)and x<y} is frequent, regardless of whether N $\cup$ CE(N) is a frequent item set. In this way, the depth-first maximum pruning strategy can well overcome the limitations of the maximum pruning strategy 3, thus greatly improving the efficiency of pruning [12].

3.3 Pruning using the parent-child relationship

If Sup(N)=Sup(N∪{i})(i∈CE(N)), then node N can be replaced by node N∪{i}. Remove other extension branches containing N but not i from the enumeration tree. When expanding the candidate space in the lexicographic subset enumeration tree, the item x must be taken out from CE(N) in order to calculate the support of N∪{x}. The deleted itemsets are all proper subsets of a maximum frequent itemset containing N∪{i}.

As shown in Figure 7, N={a}, assuming Sup({ab})=Sup(a), node a can be replaced by node a, and then delete the extension branches ac and ad.

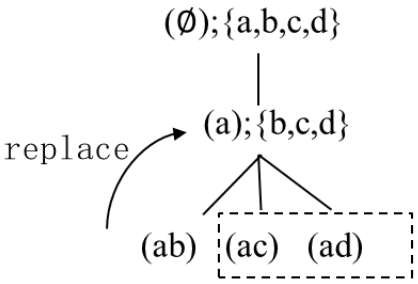

Figure 7: Prune using the parent-child relationship (the dotted box is the pruned node)

This strategy is different from other strategies. The difference is that this strategy adds support information and does not require. It is necessary to calculate other redundant things, which brings benefits and can improve the efficiency of pruning search. in addition, In the process of pruning, unlike the previous pruning strategies, it retains support information, so it can be used in frequent item set mining. Algorithms have been widely used [13].

4. Conclusion

This article mainly analyzes some issues involved in the process of mining maximum frequent itemsets. First, the relevant definitions and theoretical knowledge of frequent itemsets and maximum frequent itemsets involved in mining association rules in data mining are introduced. Then it summarizes pruning strategies to speed up search, including using infrequent subset pruning, using frequent superset pruning, and using parent-child relationship pruning. Finally, related algorithms for different pruning strategies are introduced.

References

[1] Zhang Qing, Tan Xu, Lu Xin. Research frontier and prospects of frequent item set mining [J]. Journal of Shenzhen Information Vocational and Technical College, 2024, 22(01): 1-14. 2024-01-010.

[2] Liu Wenjie, Yang Haijun. Closed frequent itemset mining algorithm based on ESCS pruning strategy [J]. Journal of Jilin University (Information Science Edition), 2023, 41(02): 329-337.2023.02.021.

[3] Gu Junhua, Li Ruting, Zhang Yajuan, et al. Research on improved frequent item set mining algorithm and its application [J]. Computer Applications and Software, 2019, 36(09): 260-269.2019-09-047.

[4] Geng Xiaohai. Research on frequent item set mining algorithm based on data flow [D]. Chongqing University of Posts and Telecommunications, 2020.2020.000163.

[5] Zhao Xincan. Research on association rules mining of high-dimensional data and incremental data based on big data environment [D]. Jiangxi University of Science and Technology, 2021.2021.000587.

[6] Zhang Yang, Wang Rui, Wu Guanfeng, et al. Parallel mining algorithm of frequent itemsets based on N-list and DiffNodeset structures [J]. Computer Science, 2023, 50(11):55-61.2023-11-008.

[7] Wang Bin, Liu Hao, Li Xiaohua, et al. High-weighted fault-tolerant frequent itemset mining algorithm based on weighted dynamic trees [J]. Journal of Qingdao University of Science and Technology, 2023, 44(03): 130-137.2023-03-017.

[8] Zhao Xuejian, Zhao Ke. Biologically inspired frequent itemset mining strategy based on genetic algorithm [J]. Computer Science, 2023, 50(S2): 636-643.
[9] Zhao Qi. Research on multi-dimensional association rule algorithm based on Fp-Growth[D]. Harbin University of Science and Technology, 2023.2023.000257.
[10] Yang Yong, Zhang Lei, Qu Fuheng, et al. THIMFUP algorithm based on most frequent item extraction and candidate set pruning [J]. Journal of Jilin University (Science Edition), 2021, 59(03): 635-642.2020277.
[11] Xu Jingwen, You Jinguo, Wang Quankun, et al. Research on the unified computing framework of data cubes and frequent itemsets [J]. Journal of Computer Science, 2023, 46(04): 780-802.2023-04-007.
[12] Shi Xiaochen. Data flow anomaly detection based on maximum frequent items [J]. Computer Knowledge and Technology, 2022, 18(25): 118-120+125.2022.1663.
[13] Yan Lixia, Ling Xinghong, Ni Hongtao. Mixed data frequent itemset mining algorithm based on Apriori algorithm [J]. Computer Simulation, 2023, 40(12): 538-542.2023-12-095.