EasyChair Preprint

№ 1367

Research on Failure Behavior Rule Base and Reasoning Engine Establishment Method

Song Yang, Ying Chen, Yaping Li and Zhengyong Liu

August 4, 2019

Research on Failure Behavior Rule Base and Reasoning Engine Establishment Method

Song Yang*[a,b] , Ying Chen[a,b]
[a]Beihang University, School of Reliability and Systems Engineering,
Beijing 100191, P.R. China
[b]Shanghai Aerospace Systems Engineering Institute
Shanghai,201109, P.R. China
1149202191@qq.com

Yaping Li, Zhengyong Liu

Shanghai Aerospace Systems Engineering Institute
Shanghai,201109, P.R. China

***Abstract* — The mining and refining methods of failure behavior rules and the establishment method of reasoning engine are studied. We proposes a mining method of related mechanisms, including the preprocessing of data based on domain knowledge, and the use of FP-growth algorithm to mine frequent item sets and correlation analysis to obtain related failure mechanism information. On this basis, the expert system technology is used to establish the failure behavior rule base, and through the iterative reasoning, all the preconditions needed to reach the final state are obtained, and the reasoning engine is established**

Keywords: FP - growth algorithm correlation analysis Failure behavior

Ⅰ INTRODUCTION

With people's deep understanding of the fundamental cause of fault, namely the failure mechanism, it has become a new idea to describe the system fault law from the perspective of fault mechanism. Under the action of external environment and load, the internal structure and material of the system may degenerate, dissipate and fail, overstress and other failure mechanisms. The fault of system - level products is the result of all kinds of fault mechanism[1-4].

Because the correlation of failure mechanism is very complex, it is usually simplified as independent or simple competitive relationship in current research. Keedy et al. [5] considered the competitive relationship between the impact overstress mechanism and fatigue wear mechanism, established the reliability model of the artificial scaffold, and proposed the maintenance strategy. Bocchetti[6] studied two main failure mechanisms of cylinder liner: wear degradation and expansion crack, and proposed a competitive risk model to predict the reliability of liner. Huang and Askin[7] studied the multi-competition fault mechanism of electronic equipment, including overstress fault mechanism and degradation mechanism, and assumed that these mechanisms were independent of each other in the reliability modeling process.

The correlation of fault mechanism describes various dynamic interactions between mechanisms, and clearly depicts the whole process of system failure. However, it is difficult to fully understand the occurrence and development rules of faults. Chen Ying et al.[8-10] summarized the correlation of mechanism layer for the first time as follows: competition, triggering, promotion, inhibition, damage accumulation and parameter combination, and described the practical significance of various relations in detail in the form of failure mechanism tree, laying the foundation for fault mechanism correlation modeling. Yang liu[11] established the fault mechanism correlation model of multi-stage mission system with Petri net, and completed the quantitative calculation of the model based on PPoF.

However, these papers only propose the corresponding model without clearly proposing an effective method, which can be used to analyze the existing fault records, maintenance records, design information of the system and thus directly obtain the correlation between the system failure mechanism. This paper proposes a mining method based on FP-growth algorithm, which solves this problem well.

The two innovations of this paper are:

1) extraction of fault behavior rules. he fault behavior rules referred to here are not abstract fault correlation relations, but study how to mine fault behavior examples from extensive failure analysis data, extract specific fault behavior rules from them, and then build a comprehensive library.

2) automatic analysis of the system. The related failure mechanism can be automatically extracted from the product

failure record, maintenance record and design information by programming

II FAILURE MECHANISM

Failure mechanism with correlation will increase the probability of component failure and reduce the service life of components so as to reduce system reliability. Therefore, the influence of failure mechanism correlation needs to be considered in the modeling of complex systems. The correlation of failure mechanism mainly includes five types: Competition, Triggering, Promotion, Inhibition and Accumulation.

A. Competition

The competition relationship of failure mechanisms can be described as that different failure mechanisms have no mutual influence in the development process, exist independently, and all have an impact on system life and reliability. FIG. 1 shows the competition failure mechanism tree and fault scene tree representation methods.

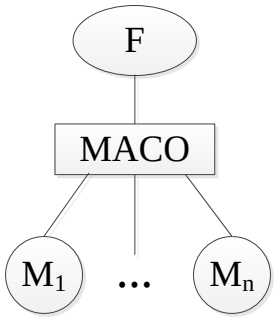

Figure 1 Failure mechanism tree and fault scenario tree of competition

The failure probability distribution function of the component is:

$$\begin{aligned} F(t) &= P(\zeta \le t) = 1 - P(\zeta \ge t) \\ &= 1 - P(\min\{\zeta_1, \zeta_2, ..., \zeta_n\} \ge t) \\ &= 1 - \prod_{i=1}^{n} P(\zeta_i \ge t) \\ &= 1 - \prod_{i=1}^{n} \left[1 - P(\zeta_i \le t)\right] \\ &= 1 - \prod_{i=1}^{n} \left[1 - \int_0^t f_i(t)dt\right] \end{aligned}$$

In the formula, ζ indicates the life of the component, ζ_i indicates the life of the component when the mechanism M_i fails alone, and $f_i(t)$ indicates the failure density function of the mechanism M_i.

B. Trigger

The triggering relationship of failure mechanism can be divided into two categories: the first is that some failure mechanism accumulates certain influence on the surrounding environment and then develops to a certain degree to trigger other fault mechanisms to transition from the potential state to the activated state and start to develop. The second is that the occurrence of a specific event triggers a failure mechanism to transition from the potential state to the activated state and start to develop. The triggered mechanism in the triggered relationship is usually a short-term destructive mechanism, and the consequences are often direct and serious. FIG 2 shows the failure mechanism tree and fault scene tree representation method of the triggering relationship.

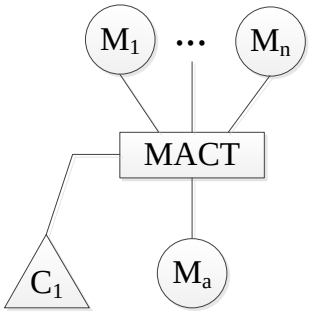

Figure 2 Failure mechanism tree and fault scenario tree of trigger relationship

The failure probability distribution function of the component is:

$$\begin{aligned} F(t) &= 1 - R(t) \\ &= 1 - P(\zeta > t) \\ &= 1 - P\left(\min\{t_a, T_C + t_1, T_C + t_2, ..., T_C + t_n\}\right) \\ &= 1 - P\left(t_a > t, T_C + t_1 > t, T_C + t_2 > t, ..., T_C + t_n > t\right) \\ &= 1 - \left[1 - P(t_a \le t)\right] \prod_{i=1}^{n} \left[1 - P(T_C + t_i \le t)\right] \\ &= 1 - \left[1 - F_a(t)\right] \prod_{i=1}^{n} \left[1 - F_i(t - T_C)\right] \\ &= 1 - \left[1 - \int_0^t f_a(t)dt\right] \prod_{i=1}^{n} \left[1 - \int_0^{t-T_C} f_i(t)dt\right] \end{aligned}$$

Where, TC represents the trigger time

C. Promotion and Inhibition

The promotion and inhibition relationship of fault mechanism is that two or more fault mechanisms change the original development rate through the mutual influence of local environment or load factors. In the promotion relationship, it is shown as accelerating the development rate of mechanism; In the inhibitory relationship, it appears to slow down the development of the mechanism. FIG. 3 shows the failure mechanism tree and the failure scenario

tree representation method of the promotion and inhibition relationship.

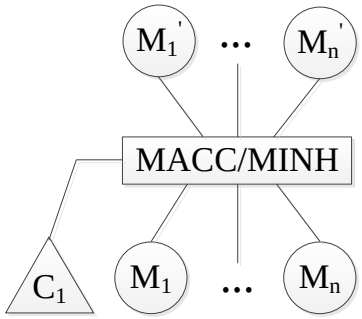

Figure 3 Failure mechanism tree and fault scenario tree of promotion and inhibition relationship

The failure probability distribution function of the component is:

$$\begin{aligned} F(t) &= P(\zeta \le t) = 1 - P(\zeta \ge t) \\ &= 1 - P(T_C + t_{r1} > t, T_C + t_{r2} > t, ..., T_C + t_{rn} > t) \\ &= 1 - \prod_{i=1}^{n} \left[1 - F_{ri}\left(t - T_C\right) \right] \\ &= 1 - \prod_{i=1}^{n} \left[1 - \int_0^{t-T_C} f_{ri}(t)dt \right] \end{aligned}$$

Where, $f_{ri}(t)$ represents the failure density function after mechanism M_i is promoted or inhibited.

D. Accumulation:

Damage accumulation describes two or more failure mechanisms that cause the same type of damage in the same part of the system. The accumulation of these damages together determines the time of system failure. FIG 4 shows the failure mechanism tree and failure scenario tree representation method of cumulative relation

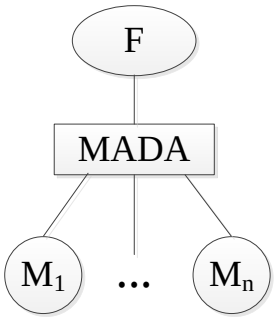

Figure 4 Failure mechanism tree and fault scenario tree of cumulative relationship

The life of the component is:

$$\begin{aligned} \zeta &= \frac{X_{th}}{\Delta X} = \frac{X_{th}}{\lambda_1 \Delta X_1 + ... + \lambda_i \Delta X_i} \\ &= \frac{X_{th}}{\lambda_1 \frac{X_{th}}{t_1} + ... + \lambda_i \frac{X_{th}}{t_i}} \\ &= \frac{1}{\frac{\lambda_1}{t_1} + ... + \frac{\lambda_n}{t_n}} = \frac{1}{\sum_{i=1}^{n} \frac{\lambda_i}{t_i}} \end{aligned}$$

X_{th} represents the threshold of the damage effect, ΔX represents the cumulative damage per unit time, ΔX_i represents the damage amount per unit time mechanism M_i, λ_i represents the proportional coefficient of the mechanism M_i, and t_i represents the component failure time when the mechanism M_i fails alone.

Then the failure probability distribution function of the component is:

$$F(\mathrm{t}) = P(\zeta \le t) = P(\frac{1}{\sum_{i=1}^{n} \frac{\lambda_i}{t_i}} \le t)$$

III THE PRINCIPLE OF APRIORI ALGORITHM

In order to explain the Apriori principle, we assume that the data set contains only four units a, b, c and d. FIG 5 shows all possible combinations of these four units. In order to find the frequent item set, the most original method is to carry out support degree judgment for each item set, and each decision needs to traverse the entire data set to find the frequency of occurrence of the item set. For a data set with only 4 kinds of cells, it also needs to be traversed 15 times. When the data set contains n kinds of cells, the traversal can reach 2^n -1 times, which will be a huge workload.

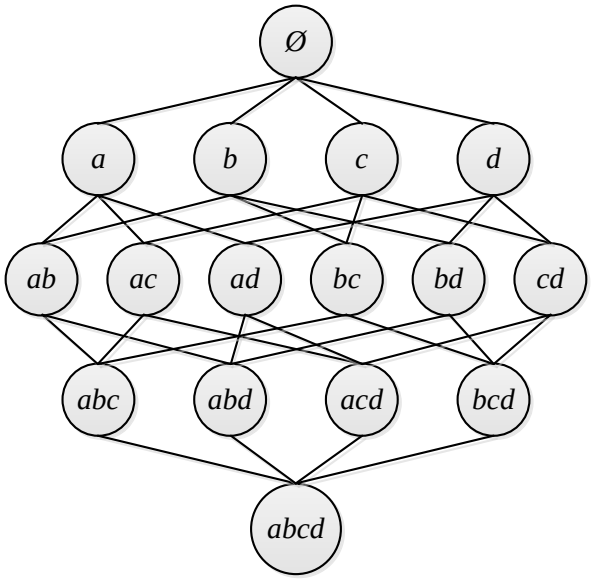

Figure 5 All possible combinations of 4 units

The Apriori principle is that if a set of items is frequent, then all its subsets are frequent, for example, given that item set {a, c} is frequent, then its subsets {a} and {c} must also be frequent. This is a fairly obvious argument, which may seem useless, but its inverse argument is very useful: if an item set is infrequent, then all of its supersets are infrequent. Given that we know item sets {b} are infrequent, all item sets containing b must be infrequent, as shown in the shaded part of figure 6.When we calculate the support degree of {b}, there is no need to calculate the support degree of other shadow item sets, which greatly reduces the number of traversal and shortens the time of the algorithm.

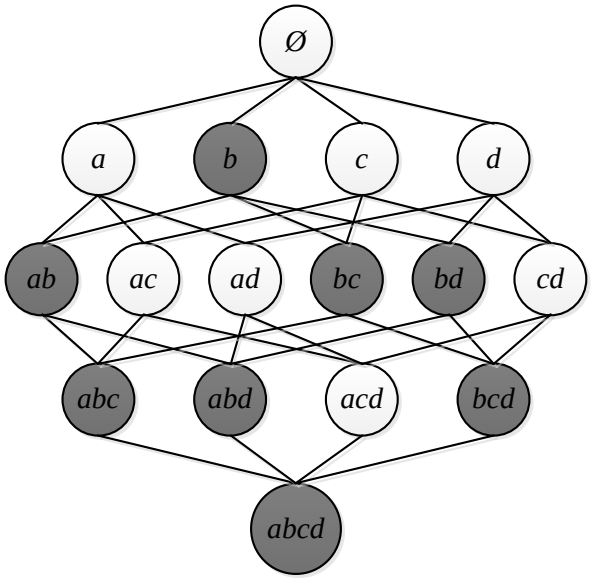

Figure 6 item set {b} is a combination of item sets that must be infrequent when infrequent

FP-growth algorithm is an improvement of Apriori algorithm, because it is more efficient in the process of mining frequent item sets. Its core idea is to build FP tree, store unit sets in a specific tree structure, and then mine frequent item sets from FP tree. The algorithm only needs to scan the data set twice, thus avoiding the combined explosion phenomenon and improving the mining efficiency.

Ⅵ CASE STUDY

A . Modeling process

The first step in modeling is to preprocess the original input data. The original pre-processing input database is the historical failure data and failure analysis report of the research object and similar objects (limited to electronic products). The data source is not limited. The file format can be Word, Excel, Txt and so on. The required domain knowledge is mainly expert analysis of the failure mechanism and failure mode of such products, which is stored in the form of keywords. The data source of the preprocessed output is the combination form and frequency of these keywords when they appear together in the original database, which will serve as the input of the association analysis algorithm. FIG 7 shows a flow chart of the pre-processing steps.

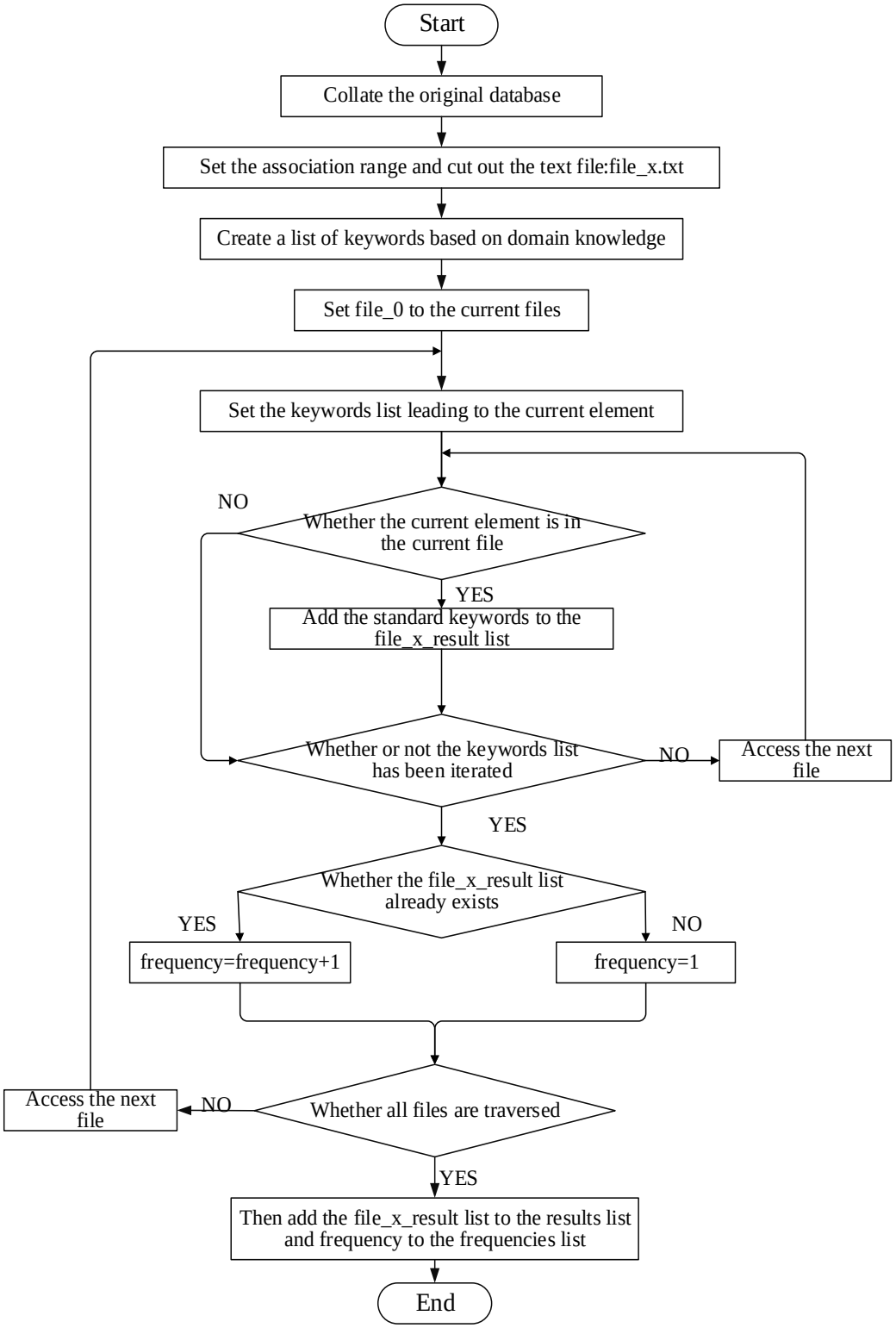

Figure 7 Electronic product failure data preprocessing flowchart

Frequencies of the standard keywords are stored in the results list obtained from the above steps. The frequencies list stores the frequencies of each of the combinations and the data will be further utilized in the subsequent associative algorithm. Manually collect and sort out the original database and domain knowledge list, cut out the text file, and then the traversal search process is completed by the program.The pseudo-code of the algorithm using Python language is as follows:

ALGORITHNM：Electronic product failure data preprocessing algorithm

INPUT：A list of text file paths ‘files’，list ‘keywords’

OUTPUT：list ‘results’，list ‘frequencies’

```
results = []
frequencies = []
for file in files:
with open(file) as file_obj:
The contents of the current file are read into the program and stored in the variable 'content'
for keyword in keywords:
for word in keyword:
  if word in content:
   Add the first element of the current 'word' list to the list 'file_x_result'
    continue
if file_x_result in results:
  Read the location of 'file x result' in the list 'results'
  Add 1 to the element at the corresponding position in list 'frequencies'
else:
  Add 'file x result' to the list 'results'
frequency = 1
  Add 'frequency' to the list 'frequencies'
return results, frequencies
```

Association analysis, also known as association rule learning, is a data mining task that searches large data sets and extracts the implicit relationships between units. There are two forms of implied relations referred to here: frequent itemsets and association rules. Frequent item set refers to the collection of units that often appear together, and association rule refers to the possible correlation between units in frequent item set. The results list and frequencies list obtained after preprocessing of data at a certain time are shown in table 1.

Table 1 keyword combination form and frequency

Num	List results	List frequencies
1	Thermal fatigue, Solder joint failure	16
2	Vibration fatigue, Solder joint failure	13
3	Thermal fatigue，Vibration fatigue，Solder joint failure	19
4	Thermal fatigue，Corrosion，Silver migration，Open	10
5	Thermal fatigue，Vibration fatigue，Electromigration，Open	6
6	Electromigration，Corrosion，Silver migration，Open	14
7	Corrosion，Silver migration，Open	13
8	Corrosion，Silver migration，Dendritic growth，Open	9

Considering that the main purpose of this paper is to mine the failure mechanism that may have an implicit relationship, and the subsequent correlation analysis of failure mechanism does not need to analyze the specific mode of fault, we use "fault" to replace all the key words of fault mode. Table 1 is updated to table 2.

Table 2 combination forms and frequency after update

NUM	Combination	Frequency
1	Thermal fatigue，Fault	16
2	Vibration fatigue，Fault	13
3	Thermal fatigue，Vibration fatigue，Fault	19
4	Thermal fatigue，Corrosion，Silver migration，Fault	10
5	Thermal fatigue，Vibration fatigue，Electromigration，Fault	6
6	Electromigration，Corrosion，Silver migration，Fault	14
7	Corrosion，Silver migration，Fault	13
8	Corrosion，Silver migration，Dendritic growth，Fault	9

As can be seen from table 2, the frequency of "thermal fatigue" and "Fault" in the same set is quite high, so it is reasonable to assume that the set {thermal fatigue, failure} is a set of frequent items. In addition, for this frequent term set, we can also deduce an association rule of "thermal fatigue failure" based on the data in the table, which means that if a component has thermal fatigue mechanism, it is likely to fail. On the contrary, the association rule "failure→ thermal fatigue" is not necessarily true, which means that if a component fails, the cause of the failure is not necessarily thermal fatigue. The assertion of association rules seems very simple and belongs to common knowledge, but the program does not have such common sense, and it needs to be analyzed by algorithm when dealing with large data sets. The degree of support is defined for a set of cells or item sets. The degree of support of an item set refers to the proportion of the combination of cells in the item set to the whole data set. Take the data in table 2 as an example, item set {thermal fatigue} appears 51 times in total, and the total number of sets is 100, so its support is 51%. For another example, item set {thermal fatigue, vibration fatigue} has 25%

support. In general, for a certain data set, the total number of item sets is fixed, so the frequency of item set can also be used to express the support degree of item set. It is usually necessary to define a minimum support, and item sets that meet the minimum support will be identified as frequent item sets. Credibility is defined against association rules. For example, the reliability of an association rule "thermal fatigue failure" is defined as "support ({thermal fatigue, failure})/ support ({thermal fatigue})".According to the data in table 2, item sets {thermal fatigue, fault} and {thermal fatigue} have 51% support, so the reliability of association rule "thermal fatigue fault" is 100%.Similarly, the reliability of "failure thermal fatigue" is only 51%, which is consistent with our previous artificial analysis results. As with support, you usually need to define a minimum level of confidence, and only association rules that meet the minimum level of confidence are preserved.

B . Building a FP tree

The construction of FP tree is the basis and core of FP-growth algorithm. As mentioned above, the construction of FP tree only needs to traverse the data set twice, and the support degree of all cells is obtained through the first traversal, which is represented by frequency. We take the data set in table 2 as an example and replace words with letter symbols. The support degree of each unit is shown in table 3.

Table 3 unit code and support

CODE	CELL	SUP
a	Thermal fatigue	51
b	Fault	100
c	Vibration fatigue	38
d	Corrosion	46
e	Silver migration	46
f	Electromigration	20
g	Dendritic growth	9

Set the minimum support to 20. It can be obtained from the above table that the support degree of set {g} is less than the minimum support degree, and according to the Apriori principle, it can be discarded directly without entering the second traversal. Therefore, we need to update table 1, replace the text with the code name, and reorder the data according to the support degree from large to small. The results are shown in table 4.

Table 4 The updated dataset

NUM	ITEMS	reordered collection	Frequency
1	*a*, *b*	*b*, *a*	16
2	*c*, *b*	*b*, *c*	13
3	*a*, *c*, *b*	*b*, *a*, *c*	19
4	*a*, *d*, *e*, *b*	*b*, *a*, *d*, *e*	10
5	*a*, *c*, *f*, *b*	*b*, *a*, *c*, *f*	6
6	*f*, *d*, *e*, *b*	*b*, *d*, *e*, *f*	14
7	*d*, *e*, *b*	*b*, *d*, *e*	13
8	*d*, *e*, *g*, *b*	*b*, *d*, *e*	9

The second traversal is the process of tree building, and the traversed data set is the filtered and reordered collection. From an empty set ϕ start reading in each set, and will be read in a collection of added to the tree, if existing the branches of trees, increase the value of each element; If some elements are consistent with an existing branch, a new element is added after that branch. If there are no branches that overlap an existing element, a new one is added to the tree. The flow of building FP trees for the datasets in table 4 is shown in figure 8.

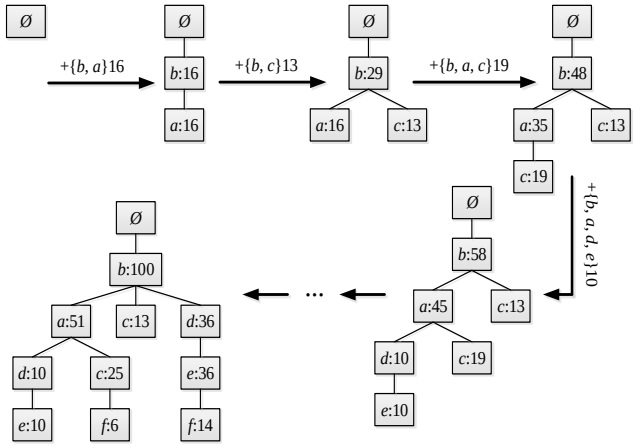

Figure 8 FP tree construction process

In addition, a table of head Pointers is also required to achieve fast access to elements in the FP tree. The table of head Pointers stores the total number of each element, and points the pointer to the first instance of each element in the FP tree, and then the first instance points to the next instance until all the same elements are connected. The FP tree of the lead pointer table is shown in figure 9.

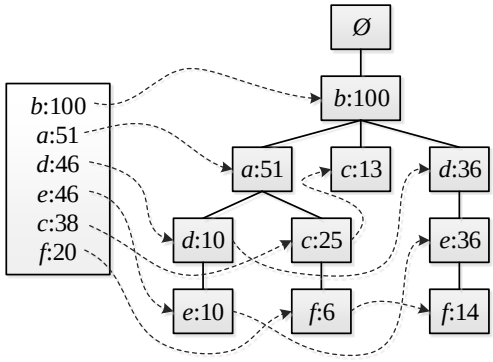

Figure 9 FP tree with head pointer table

The general function code of using Python language to build FP tree is as follows:

ALGORITHM：FP-growth algorithm

INPUT：data_set and min_sup

OUTPUT：fp_tree and head_table

```
def create_tree(data_set, min_sup=20):
        '''Create FP TREE'''
        head_table = {}
        for sets in data_set:
                for item in sets:
        head_table[item] = head_table.get(item, 0) + data_set[sets]
        for k in head_table.keys():
                if head_table[k] < min_sup:
                        del(head_table[k])
        freq_itemset = set(head_table.keys())
        if len(freq_itemset) == 0:
                return None, None
        else:
                for k in head_table:
                        head_table[k] = [head_table[k], None]
                fp_tree = Tree('Null Set', 1, None)
                for sets, count in data_set.items():
                        local = {}
                        for item in sets:
                                if item in freq_itemset:
                        local[item] = head_table[item][0]
                        if len(local) > 0:
                order_items = [v[0] for v in sorted(local.items(),
                        key=lambda p: p[1], reverse=True)]
                update_tree(order_items, fp_tree, head_table, count)
        return fp_tree, head_table
def update_tree(items, fp_tree, head_table, count):
        if items[0] in fp_tree.children:
                fp_tree.children[items[0]].inc(count)
        else:
                fp_tree.children[items[0]] = Tree(items[0], count,
fp_tree)
                if head_table[items[0]][1] == None:
                head_table[items[0]][1] = fp_tree.children[items[0]]
                else:
                        update_head(head_table[items[0]][1],
                                fp_tree.children[items[0]])
        if len(items) > 1:
                update_tree(items[1::], fp_tree.children[items[0]],
                        head_table, count)
def update_head(node, target):
        while node.nodelink != None:
                node = node.nodelink
        node.nodelink = target
```

C . Mining frequent itemsets from FP trees

After the FP tree is constructed, there is no need to traverse the data set again. Frequent item sets will be inferred directly from the FP tree, as follows:

1) read conditional mode base from FP tree;

2) construct conditional FP tree according to conditional pattern basis;

3) extract frequent item sets from the conditional FP tree;

4) iterate steps 1) through 3) until the conditional FP tree is empty.

The conditional schema base is defined for the item set. Since we constructed the FP tree in descending order of element support, the conditional schema base can be defined as all prefix paths of the item set to be looked up, and prefix paths refer to all branch information between the item set to be looked up and the root node. Table 5 shows all the prefix paths for the singleton itemset, which is determined to be frequent.

Table 5 Prefix path for a single element item set

Single element item set	Prefix path
a	{*b*}51
b	{}100
c	{*b*，*a*}25，{*b*}13
d	{*b*，*a*}10，{*b*}36
e	{*b*，*a*，*d*}10，{*b*，*d*}36
f	{*b*，*a*，*c*}6，{*b*，*d*，*e*}14

The number after the prefix path set is the frequency of the prefix path. With the conditional schema base for the singleton itemset determined, a conditional FP tree can be created. Take element 'e' as an example, the component flow is shown in figure 10.

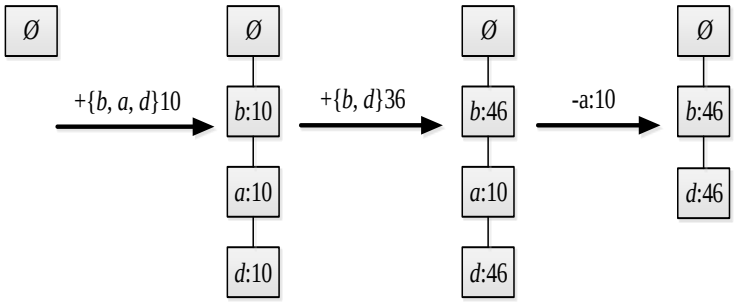

Figure 10 element e conditional FP tree component flow

In the construction process, the conditional pattern basis of element e is added one by one, and the elements that do not meet the minimum support degree in the tree are removed, and it is obtained that element e is related to element b and d respectively, that is, item set {e, d} and {e, b} are frequent, and then the conditional FP tree of these two item sets is constructed as shown in FIG 11.

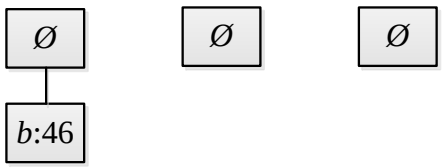

Figure 11 conditional FP tree of item sets {e, d}, {e, b} and {e, d, b}

Item set {e, d} contains element b in the conditional FP tree, indicating item set {e, d, b} is also frequent, while item set {e, b} and {e, d, b} end when the conditional FP tree is null. Similarly, other frequent item sets can be obtained, and all frequent item sets of this example are finally obtained, as shown in table 6

Table 6 all frequent itemsets

Frequent itemsets				
$\{a\}$	$\{b\}$	$\{c\}$	$\{d\}$	$\{e\}$
$\{f\}$	$\{a, b\}$	$\{a, c\}$	$\{b, c\}$	$\{a, b, c\}$
$\{b, d\}$	$\{b, e\}$	$\{d, e\}$	$\{b, d, e\}$	$\{b, f\}$

The code name 'b' stands for "failure", which belongs to the failure mode. If we want to investigate which mechanism may have an implicit relationship, we can remove the element b in the above table and remove the single element item set, and the mechanism set that may have an implicit relationship is {thermal fatigue, vibration fatigue} and {corrosion, silver migration}.The general function code to mine frequent item sets from FP tree by Python language is as follows:

ALGORITHM：FP-growth algorithm

INPUT：fp_tree、head_table and min_sup

OUTPUT：freq_itemlist

```
def ascend_tree(leaf_node, prefix_path):
        if leaf_node.parent != None:
                prefix_path.append(leaf_node.name)
                ascend_tree(leaf_node.parent, prefix_path)
def find_prefix_path(base_pat, tree_node):
        cond_pats = {}
        while tree_node != None:
                prefix_path = []
                ascend_tree(tree_node, prefix_path)
                if len(prefix_path) > 1:
                        cond_pats[frozenset(prefix_path[1:])] = tree_node.count
                tree_node = tree_node.nodelink
        return cond_pats
def mine_tree(fp_tree, head_table, min_sup, prefix, freq_itemlist):
        bigL = [v[0] for v in sorted(head_table.items(), key=lambda p: p[1])]
        for base_pat in bigL:
                new_freq_set = prefix.copy()
                new_freq_set.add(base_pat)
                freq_itemlist.append(new_freq_set)
                cond_patt_bases = find_prefix_path(base_pat, head_table[base_pat][1])
                my_cond_tree, my_head = create_tree(cond_patt_bases, min_sup)
                if my_head != None:
                        mine_tree(my_cond_tree, my_head, min_sup, new_freq_set, freq_itemlist)
        return freq_itemlist
```

Through the case of correlation analysis, we can extract the key fault information from a large number of maintenance records, failure reports and design information of the product, and conduct correlation analysis on the information to obtain the coupling state of the product fault. It is necessary and significant to carry out the correlation analysis of the product failure mechanism because the failure of the product is often not a single failure.

This method firstly uses data mining technology to mine the related fault mechanism, including data preprocessing based on domain knowledge and frequent item sets mining by FP-growth algorithm for association analysis, breaking the deadlock of previous association analysis relying on expert knowledge and experience, and realizing automatic program analysis.

V ACKNOWLEDGEMENT

This work was funded by the Defense Industrial Technology Development Program of China (grant numbers JSZL2015601B007)

[1] 赵广燕. 系统故障行为建模方法研究[D]. 北京:北京航空航天大学, 2006.5

[2] Zhao G. and Zhao G. System fault behavior model [A]. The Proceedings of 2009 8th International Conference on Reliability, Maintainability and Safety [C]. 2009: 925-929

[3] 查国清, 井海龙, 陈云霞, 等. 基于故障行为模型的产品寿命分析方法[J]. 北京航空航天大学学报, 2016, 42(11): 2371-2377

[4] Zeng Z. and Kang R. Failure behavior modeling: Towards a better characterization of product failures [J]. Chemical Engineering, 2013, 33: 571-576

[5] Keedy E. and Feng Q. A Physics-of-failure based reliability and maintenance modeling framework for stent deployment and operation [J]. Reliability Engineering and System Safety, 2012, 103: 94-101

[6] Bocchetti D., Giorgio M., Guida M., et al. A competing risk model for the reliability of cylinder liners in marine Diesel engines [J]. Reliability Engineering and System Safety,2009, 94(8): 1299-1307

[7] Huang W. and Ronald G. Reliability analysis of electronic devices with multiple competing failure modes involving performance aging degradation [J]. Quality and Reliability Engineering International, 2003, 19(3): 241-254

[8] Chen Y., Yang L., Ye C., et al. Failure mechanism dependence and reliability evaluation of non-repairable system [J]. Reliability Engineering and System Safety, 2015, 138:273-283

[9] 袁增辉. 焊点不规则温循热疲劳损伤累积规律研究[D]. 北京: 北京航空航天大学,2016

[10] Li Y. Y., Chen Y., Yuan Z. H., et al. Reliability analysis of multi-state systems subject to failure mechanism dependence based on a combination method [J]. Reliability Engineering and System Safety, 2016

[11] 杨柳. 基于 PPoF 的多阶段任务系统可靠性建模方法研究[D]. 北京:北京航空航天大学, 2011