Efficient Mining Maximal Variant Usage and Low Usage Biclusters in Discrete Function-Resource Matrix

Lihua Zhang1,2, Miao Wang2,3,*, Zhengjun Zhai1, Guoqing Wang1,2,3
1School of Computer Science and Engineering, Northwestern Polytechnical University, Xi’an, China, 710072
2Science and Technology on Avionics Integration Laboratory, Shanghai, China, 200233
3China National Aeronautical Radio Electronics Research Institute, Shanghai, China, 200233
Email: {zhang_lihua, wang_miao,wang_guoqing}@careri.com
*Corresponding author

Abstract—The functional layer is the pillar of the whole prognostics and health management system. Its effectiveness is the core of system task effectiveness. In this paper, we proposed a new bicluster mining algorithm: DoCluster, to effectively mine all biclusters with maximal variant usage rate and low usage rate in the discrete function-resource matrix. In order to improve the mining efficiency, DoCluster algorithm constructs a sample weighted graph firstly; secondly, all biclusters with maximal variant usage rate and low usage rate satisfying the variant usage rate and low usage rate definition are mined using sample-growth and depth-first method in the constructed weighted graph. DoCluster algorithm also uses several pruning strategies to ensure the mining of maximal bicluster without candidate maintenance. The experimental results show DoCluster algorithm is more efficient than other two algorithms.

Index Terms—bicluster, variant usage rate, low usage rate, function, resource

I. INTRODUCTION

The function is the foundation of task realization and also the basis of improving and guaranteeing quality, performance and effectiveness of system task information. The functional layer is the pillar of the whole system. Its effectiveness is the core of system task effectiveness. The health of the functional layer includes the status of functional components in the hierarchy range and overall health status of the whole functional layer. Health management objective of the functional layer is the effectiveness of the functional components and the hierarchy and to form function self-organizing platform based on the effectives of functional components. Although studying the effectiveness degree of resources is the base to construct a prediction and health management system [1]. The health degree of resources directly influences functional health. So, analysis of the call relation between functions and resources can excavate the health relation between them so as to complete the functions through using healthy resources and improve the health degree of functions.

The call relation of functions and resources can be abstracted as a matrix. In other words, each row means a resource and each column means a function, the value in the matrix is the use degree of a function to a resource. This value is defined during functional design, i.e. resource dependence degree of this function in aircraft system in order to complete a function. For example, for the resource whose storage spaces are 100K, function F1 needs 60K storage spaces to store some temporary variables. The dependence degree of this function on this storage resource is 0.6. Through mining the above function-resource matrix, the usage relation between a group of functions and a group of resources can be gained. For instance, for a group of functions F1, F2, F3, the resource relations called by each function are as follows: F1 => R1R2R3, F2 => R2R4R5 and F3 => R6R7. Suppose F1, F2, F3 need to cooperate to complete a task T. All above three functions may be called at the same time. For resource R5, it supports F2 and F3 simultaneously. There may have two conditions: (1) R2 has high effectiveness for F1, but has low effectiveness for F2 and F3; (2) R2 has high effectiveness for both F1 and F2. The health degree of the first condition is higher than that of the second one. The reason is that, resource R2 can serve F1 and F2 simultaneously in the first condition; while in the second condition, resource R2 needs to serve for two functions. From the perspective of functional health, if resource R2 has defects, its influence on the first condition is lower than the second one. So, through function-resource matrix mining, in order to achieve a group of functions, the resources which can satisfy all functional demands simultaneously and the resources which can satisfy all functional demands through multiple accesses can be mined, i.e. mine bicluster with variant usage rate or low usage rate from function-resource matrix.

The above mining concept complies with the bicluster in data mining field. Biclustering concept was first proposed by Cheng and Church [2]. As a special clustering method [3-9], bicluster does not generate cluster in overall experimental conditions, but only finds out the item sets with special significance for specific matrix sample. Thus, biclustering algorithm can mine...
bicluster with variant usage rate and low usage rate described above from function-resource matrix. Currently, large quantities of algorithms based on greedy strategy or exploratory strategy are applied in mining bicluster. Cheng and Church proposed an algorithm based on greedy strategy [2]. This algorithm adopts a low square root residue to delete redundant nodes step by step. After that, many algorithms based on greedy strategy were raised [10-17]. All the above algorithms adopt the following two mining strategies: 1) produce cluster overall according to traditional clustering method and then optimize gradually; 2) mine bicluster in two types of data respectively and then gain the result through comparison and integration. But for the above two strategies, the efficiency of algorithms are not well. Thus, to design a high-efficiency bicluster mining algorithm is current research hotspot. So, Wang et al. came up with the mining algorithm to mine the maximal bicluster in discretized data [18].

The existing differential bicluster mining methods can be classified into two groups. One is to construct a difference matrix to mine discriminative biclusters. [19] developed a methodology for differential co-expression on a global scale. [20] proposed an algorithm to extract differential biclusters from the two gene expression datasets. [21] aims to mine subspace differential co-expression patterns. And it can also be used for mining differential biclusters. Another recent proposed algorithm called DeBi [22] uses frequent pattern mining approach for discovering maximum size homogeneous bicluster in which all genes are co-expressed under a subset of samples. However, this algorithm cannot effectively mine bicluster with variant usage rate meeting difference restraint from function-resource matrix.

We can see through the above analysis that existing bicluster algorithm has some shortcomings during mining a bicluster with variant usage rate and with low usage rate. In order to improve mining efficiency, this paper proposed a new bicluster mining algorithm - DoCluster algorithm which can effectively mine all biclusters with maximal variant usage rate and low usage rate from discrete function-resource matrix. Since the number of functions is far lower than that of resources in function-resource matrix, this algorithm uses sample-growth method for mining. First, a sample weighted graph is constructed, which includes all resource collections between both samples that satisfy the definition of variant usage rate or low usage rate; then, all biclusters with maximal variant usage rate and low usage rate satisfying the definition are mined with the mining method of using depth-first sample-growth method in the weighted graph. To improve the mining efficiency of the algorithm, DoCluster algorithm uses several pruning strategies to ensure the mining of maximal bicluster without candidate maintenance.

II. PROBLEM DESCRIPTION

Function-resource matrix is defined as a two-dimensional real matrix \( D = R \times F \), in which row set \( R \) represents the set of resources and column set \( F \) refers to the set of functions. Element \( D_{ij} \) of matrix \( D \) is a real number which represents the ability validity or usage rate of resource \( i \) supporting function \( j \). \(|R|\) is the number of resources in data set \( D \) and \(|F|\) is the number of functions in data set \( D \). For the convenience of mining, the original effective values in function-resource matrix are usually dispersed as 1, -1 and 0, where -1 means the usage rate of the resource is the minimum during the implementation of some function; 0 means the usage rate of the resource is moderate during the implementation of some function; 1 means the usage rate of the resource is the maximal during the implementation of some function, as shown in Table 1.

The significance of bicluster to be mined from function-resource matrix as shown in Table 1 is to mine a group of functions executed; under this group of functions, the usage rate of the resource is the maximal, i.e. which resources can reach the maximal usage rate when used together. In other words, the resources have the highest effectiveness when all functions are executed. For example, for a group of functions \( F_1, F_2 \): (1) for \( F_1 \), the usage rate of \( R_3 \) is high, while it is low for \( F_2 \), as shown in Table 2; (2) for both \( F_1 \) and \( F_2 \), the usage rate of \( R_3 \) is high, as shown in Table 3; (3) for both \( F_1 \) and \( F_2 \), the usage rate of \( R_3 \) is low, as shown in Table 4, the health degree in the first and the third conditions is higher than the second condition. The reason is that \( R_3 \) can serve \( F_1 \) and \( F_2 \) at the same time in the first and the third conditions resource. In the third condition, resource \( R_3 \) needs to serve the two functions respectively. This paper puts forward that bicluster mined by DoCluster algorithm aims at the first and third conditions.

**TABLE I.**

AN EXAMPLE OF FUNCTION-RESOURCE MATRIX

<table>
<thead>
<tr>
<th></th>
<th>( F_1 )</th>
<th>( F_2 )</th>
<th>( F_3 )</th>
<th>( F_4 )</th>
</tr>
</thead>
<tbody>
<tr>
<td>( R_1 )</td>
<td>1</td>
<td>-1</td>
<td>-1</td>
<td>-1</td>
</tr>
<tr>
<td>( R_2 )</td>
<td>-1</td>
<td>1</td>
<td>-1</td>
<td>-1</td>
</tr>
<tr>
<td>( R_3 )</td>
<td>1</td>
<td>-1</td>
<td>-1</td>
<td>-1</td>
</tr>
<tr>
<td>( R_4 )</td>
<td>0</td>
<td>1</td>
<td>-1</td>
<td>-1</td>
</tr>
</tbody>
</table>

**TABLE II.**

AN EXAMPLE OF VARIANT USAGE RATE

<table>
<thead>
<tr>
<th></th>
<th>( F_1 )</th>
<th>( F_2 )</th>
</tr>
</thead>
<tbody>
<tr>
<td>( R_1 )</td>
<td>1</td>
<td>-1</td>
</tr>
<tr>
<td>( R_2 )</td>
<td>1</td>
<td>-1</td>
</tr>
<tr>
<td>( R_3 )</td>
<td>1</td>
<td>0</td>
</tr>
<tr>
<td>( R_4 )</td>
<td>0</td>
<td>-1</td>
</tr>
</tbody>
</table>

**TABLE III.**

AN EXAMPLE OF NON-VARIANT USAGE RATE

<table>
<thead>
<tr>
<th></th>
<th>( F_1 )</th>
<th>( F_2 )</th>
</tr>
</thead>
<tbody>
<tr>
<td>( R_1 )</td>
<td>1</td>
<td>-1</td>
</tr>
<tr>
<td>( R_2 )</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>( R_3 )</td>
<td>1</td>
<td>0</td>
</tr>
<tr>
<td>( R_4 )</td>
<td>0</td>
<td>-1</td>
</tr>
</tbody>
</table>
The weighted graph can be expressed as $GE = (V, W)$, where $V$ is the vertex set, and $W$ is the weight function. Each edge in the weighted graph is denoted as $V_iV_j$, and the weight under $V_iV_j$ is $W_{ij}$.

**Definition 1.** In order to facilitate description of bicluster with variant usage rate and low usage rate, suppose the use values of resource $R_j$ after discretization under the functions $F_i$ and $F_j$ are $V_j$ and $V_j'$. There are four representations for $R_j$ under $F_i$ and $F_j$: (1) if $V_j = 1$ and $V_j' = 0$, or $V_j = 0$ and $V_j' = 1$, the contribution rate of $R_j$ to $F_i$ and $F_j$ satisfies diversity requirement, expressed as $'R_j'$ and $'*R_j'$ respectively; (2) if $V_j = 0$ and $V_j' = 0$, the contribution rate of $R_j$ to $F_i$ and $F_j$ satisfies diversity requirement, expressed as $'-R_j'$; (3) if $V_j = 1$ and $V_j' = 1$, the contribution rate of $R_j$ to $F_i$ and $F_j$ does not satisfy diversity requirement, so no record is given; (4) if $V_j = 0$ or $V_j' = 0$, the contribution rate of $R_j$ to $F_i$ and $F_j$ does not meet diversity requirement, so no record is given.

Thus, in bicluster mined by DoCluster algorithm, each resource can satisfy the first or the second conditions described above under all functions. To improve mining efficiency of the algorithm, DoCluster algorithm mines biclusters with maximal variant usage rate and maximal low usage rate by using sample-growth method without candidate maintenance. The mining process of this algorithm will be introduced in the next section.

### III. THE DOCLUSTER ALGORITHM

The mining steps of DoCluster algorithm can be divided into two steps: firstly, scan original function-resource matrix, according to the definition of biclusters with maximal variant usage rate and maximal low usage rate, all sample weighted graphs satisfying the above definition are produced; then, use sample-growth method to mine all biclusters with maximal variant usage rate bicluster and maximal low usage rate bicluster.

#### A. Construct Sample Relational Weighted Graph

The method of mining modes with sample relational weighted graph was used in MicroCluster algorithm [12] to mine bicluster firstly. Then, Wang et al. [18, 23] also used sample relational weighted graph to mine bicluster and fault-tolerant bicluster. DoCluster algorithm in this paper will adopt undirected sample relational weighted graph (hereinafter referred to as sample weighted graph) to mine biclusters with maximal variant usage rate and maximal low usage rate.

**Definition 2.** Sample weighted graph can be expressed with the set $G = \{E, V, W\}$. Each node in the vertex set $V$ in the weighted graph represents a function. If an edge exists between a pair of vertices, this means the resource with variant usage rate or low usage rate exists below two functions represented by this pair of vertices. The set of the edges is denoted as $E$. The weights of each edge are the resource set satisfying the definition of variant usage rate or the definition of low usage rate under the two functions connected with this edge. The set of the weights is denoted as $W$.

According to the description in Definition 1, when the resources among functions satisfy the definition of variant usage rate, the weight between two functions does not satisfy commutativity. For instance, the weight under $F_iF_j$ is $R_j*R_i*R_j$, while the weight under $F_jF_i$ is $R_i*R_j*R_j$. So, in Definition 2, the weight of each edge is the weight under $F_iF_j$, where $i \neq j$. Fig.1 shows the weighted graph corresponding to Table 1. For the convenience of follow-up description, Fig.2 provides storage structure of Fig.1.

![Figure 1. The sample weighted graph constructed from Table 1](image1)

![Figure 2. The storage structure of Fig.1](image2)

### B. Mining Maximal Bicluster

After the sample weighted graph is constructed, this section will introduce how DoCluster algorithm mines all biclusters with maximal variant usage rate and maximal low usage rate from sample weighted graph without candidate maintenance in detail. According to the description in Definition 2, biclusters with variant usage rate and low usage rate extended satisfy anti-monotonicity, i.e. if the bicluster obtained by extension of $F_1F_2...F_n$ does not satisfy constraint conditions, neither does any superset $F_1F_2...F_kF_{k+1}...F_n$. Therefore, biclusters with a greater scale can be obtained by extension of the weight on each edge in the weighted graph in terms of intersection. But the bicluster mined by DoCluster is different from the extension mode described in [18]. In FDCluster algorithm, if $S_1$ is gained through extending $S_1S_2$, $S_1S_3$ set can be obtained through calculating the intersection of the weight of $S_1S_2$ and the weight of $S_1S_3$. However, this scheme cannot be used in this algorithm. It is required to calculate the intersection of the edges of $F_iF_j, F_iF_j$ and $F_iF_j$ in order to gain the resource set meeting conditions under $F_iF_j$. Only in this way, such situation can be avoided that two or more ‘1’ occur
simultaneously. For example, in Table 1, for resource \( R_j \), when extended to \( F_j \) from \( F_1,F_2,F_3,F_4 \), \( R_j \) is included in the weights under \( F_1,F_2,F_3,F_4 \). However, \( R_j \) is not included in the weights under \( F_1,F_2 \). If the intersection of \( R_j \) and the weight of \( F_1,F_2 \) is not calculated when extended to \( F_3,F_4 \) from \( F_1,F_2,F_3,F_4 \), a wrong \( F_1,F_2,F_3,F_4 \) bicluster including \( R_j \) will emerge. So, when a new function is introduced in bicluster, it is necessary to calculate the intersection of all edges of the function newly introduced and the resource collection of bicluster extended. When calculating the intersection of the weights, it is only necessary to calculate the intersection of the resources, not necessary to consider ‘*’ or ‘-’ symbols before resources. With different symbols before resources, the intersection can also be calculated. These symbols are only used in pruning design.

We will introduce how DoCluster algorithm uses pruning strategies to mine all biclusters with maximal variant usage rate and maximal low usage rate from sample relationship weight graph without candidate maintenance in detail. This paper will judge maximal bicluster with the method of backward checking proposed in [24] without candidate maintenance. That is to say, if resources under the current candidate sample and some prior candidate sample (mined sample) have some inclusion relation, i.e. all biclusters produced by the current candidate sample can be produced by some prior candidate sample, the current candidate sample can be pruned. When calculating the intersection of the weights, it is just necessary to calculate the intersection of resources, and the intersection can also be calculated with different symbols before resources. But, in accordance with the description (1) in Definition 1, for resource expression forms of \( V_i \) and \( V_j \) or \( V_i \) and \( V_j \), resource expression forms under \( F_1,F_2 \) and \( F_1,F_3 \) may be different. For example, when mining \( F_1,F_2 \), the candidate functions are \( F_1(-R_3,R_2,R_5,R_2) \), \( F_1(-R_3,R_2,R_5,R_2) \), \( F_1(R_1,R_2,R_5,R_2) \), and \( F_1(-R_3,R_2,R_5,R_2) \), and the prior candidate function is \( F_1(R_1,R_2,R_5,R_2) \). Since currently \( F_1,F_2 \) is extended, \( F_1,F_2 \) is its prior candidate function. At this moment, \( F_1,F_2 \) should be produced, instead of \( F_1,F_2 \). As resource expression forms under \( F_1,F_2 \) and \( F_1,F_3 \) are different, the weighted graph made by this algorithm is a directed graph rather than undirected graph. For \( F_a,F_b \) and \( F_b,F_a \), it is necessary to build edges on \( F_a,F_b,F_a,F_b \) respectively. For \( F_a,F_b \) and \( F_b,F_a \), the difference of weights on the edge is the interchange of resource expression forms “\( R_i \)” and “\( \sim R_i \)”. Therefore, for saving the storage space, the storage of weight is only that of weight on \( F_{i,j} \) edge. The weight on \( F_{i,j} \) edge can be calculated with \( F_{i,j} \). For instance, the storage structure of Table 1 is as shown in Fig.2. \( F_1,F_2 \) can be gained through “complementing” \( F_1,F_2 \) (“\( R_i \)” and “\( \sim R_i \)” interchange, and “\( \sim R_i \)” remains unchanged).

During function extension, the resource “symbol” is not considered. But during candidate function pruning, it is necessary to judge according to resource symbols under the candidate functions. Here, resource symbols under the candidate functions are decided by candidate functions at current layer and resource symbols of the weights on the edge of initial extension function. For example, according to the storage structure shown in Fig.2, assuming the bicluster extended currently is \( F_1,F_2 \), its candidate functions are \( F_1, (-R_3,R_2,R_5,R_2) \), \( F_1 \)’s prior candidate function is \( F_1 \). Resource \( (-R_3,R_2,R_5,R_2) \) under candidate function \( F_1 \) is gained through calculating the intersection of the weights of \( F_3,F_4,F_5,F_6 \) and \( F_1,F_2,F_3 \). The “symbol” of each resource is the resource “symbol” on the edge \( F_1,F_2 \). Because function \( F_2 \) is extended currently, resource symbols of candidate functions are decided by \( F_2,F_3 \). Similarly, for prior candidate function \( F_1 \) of \( F_2,F_3 \), its resource is also gained through calculating the intersection of \( F_4,F_5,F_6 \) and \( F_2,F_3 \). Its resource symbols are decided by resource symbols on \( F_2,F_3 \).

| TABLE V. AN EXAMPLE OF PRUNING USED MATRIX |
|----|----|----|----|
| R1 | F1 | -1 | -1 | -1 |
| R2 | -1 | 1  | -1 | -1 |
| R3 | 1  | -1 | -1 | -1 |

Resource \( R_j \) is respectively expressed as ‘\( R_j \)’ and ‘\( \sim R_j \)’ above when the form of expression of resources is illustrated, just for the convenience of design of pruning strategies. For example, assuming there is the sole resource \( R_j \) in Table 5, for \( R_j \), when the extension starts from \( F_1,F_2 \), according to the above description, the expression form of \( R_j \) is ‘\( R_j \)’. Assuming all functions extended from \( F_1,F_2 \) have been extended, when extending \( F_1,F_3 \), the expression form of \( R_j \) on the edge of \( F_1,F_3 \) is also ‘\( R_j \)’. At this moment, \( F_1,F_3 \) can be pruned. It is known from the expression form of \( R_j \) that ‘1’ must exist under \( F_1,F_3 \). According to previous variance definition, ‘1’ impossibility exists under other functions extended from \( F_1,F_3 \). That is to say, \( R_j \) can only be ‘1’ under \( F_1,F_3 \). Therefore, functions which \( F_1,F_3 \) can extend must be gained through \( F_1,F_3 \) extension. Meanwhile, \( F_1,F_3 \) can extend \( F_1,F_3 \). So, \( F_1,F_3 \) can be pruned.

If a resource in the current candidate function to be extended meets the form of ‘\( R_j \)’, this resource can be pruned according to the Lemma 1 below.

**Lemma 1.** Assuming that \( P \) is the bicluster with variant usage rate to be extended currently; \( M \) is the candidate function set of \( P \) and \( N \) is the prior candidate function set of \( P \). If the expression form is ‘\( R_j \)’ for any resource \( R_j \) in candidate function \( M \) (\( M \in M \)) and there is a prior candidate function \( N_j \) (\( N_j \in N \)) under which resource \( R_j \) also exists, resource \( R_j \) in \( M \) can be obtained by extension of prior candidate function \( N_j \).

**Proof.** Proof by contradiction is adopted. Resource expression form of current candidate function \( M \) is ‘\( R_j \)’; a prior candidate function \( N_j \) (\( N_j \in N \)) exists; resource \( R_j \) also exists under \( N_j \). Thus, \( M \) can be pruned. In line with description (1) in Definition 1, for resource \( R_j \), ‘1’ is under some function in \( P \). In accordance with the definitions of variant usage rate and low usage rate,
resource $R_j$ must be ‘-1’ under candidate function $M_i$ and prior candidate function $N_j$. So, the bicluster extended currently must be a bicluster with variant usage rate. As only one ‘1’ can exist for each resource under all functions in the bicluster with variant usage rate, the bicluster with variant usage rate gained through extension of $PM$ can be obtained through extension of $PN/M_i$. Thus, $M_i$ can be pruned. This contradicts the assumption, so the original proof is established.

However, for the expression form of ‘*’, the above pruning strategy is not applicable. For example, assuming there is the sole resource $R_2$ in Table 5, for $R_2$, when the extension starts from $F_iF_2$, according to the previous description, the expression form of $R_2$ is ‘*R’$. Assuming all functions extended from $F_iF_2$ have been extended, when extending $F_iF_2$, the expression form of $R_2$ on the edge of $F_iF_2$ is also ‘-R’$. At this moment, $F_iF_2$ cannot be pruned. It is known from the expression form of $R_2$ that ‘1’ must exist under $F_2$. According to previous variance definition, ‘1’ likely exists under other functions extended from $F_i$. That is to say, ‘1’ likely appears under the functions extended by $F_iF_1$. For $R_2$, $F_iF_2F_3F_4$ can be gained through extension of $F_iF_2$, but $F_iF_2F_3F_4F_5$ cannot be gained through extension of $F_iF_2$. Therefore, functions which $F_iF_2$ can extend may not be gained through $F_iF_2$ extension. So, For $R_2$, $F_iF_2$ cannot be pruned.

According to the above analysis, if a resource in the current candidate function to be extended satisfies the form of ‘*R’$, it should be judged whether this resource can be pruned according to the weight of prior candidate function. Therefore, the following Lemma can be used for pruning.

**Lemma 2:** assuming that $P$ is the bicluster with variant usage rate to be extended currently; $M$ is the candidate function set of $P$ and $N$ is the prior candidate function set of $P$. If the expression form is ‘*R’$ for any resource $R_j$ in candidate function $M_i$ ($M_i \in M$) and there is a prior candidate function $N_j$ ($N_j \in N$) under which resource $R_j$ with the expression form of ‘-R’$ also exists, resource $R_j$ in $M_i$ can be obtained by extension of prior candidate function $N_j$.

**Proof:** Proof by contradiction is adopted. When resource expression form of current candidate function $M_i$ is ‘R’$, a prior candidate function $N_j$ ($N_j \in N$) exists; resource $R_j$ also exists under $N_j$ with the expression form of ‘-R’$, $M_i$ can be pruned. In line with description (1) in Definition 1, for resource $R_j$, ‘1’ is under current candidate function in $M_i$. In accordance with the definitions of variant usage rate and low usage rate, resource $R_j$ under all functions in $P$ must be ‘1’. Since resource $R_j$ also exists under $N_j$ with the expression form of ‘-R’$, the bicluster extended currently must be a bicluster with low usage rate. As only one 1 can exist for each resource in the bicluster with variant usage rate, the bicluster $PN/M_i$ with variant usage rate can be gained through extension of $PN$. Thus, the bicluster with variant usage rate gained through extension of $PM$ can be obtained through extension of $PN/M_i$. Thus, $M_i$ can be pruned. This contradicts the assumption, so the original proof is established.

Similarly, if a resource in the current candidate function to be extended meets the form of ‘-R’$, it should be judged whether this resource can be pruned according to the weight of prior candidate function. Therefore, the following Lemma can be used for pruning.

**Lemma 3:** assuming that $P$ is the bicluster with variant usage rate to be extended currently; $M$ is the candidate function set of $P$ and $N$ is the prior candidate function set of $P$. If the expression form is ‘-R’$ for any resource $R_j$ in candidate function $M_i$ ($M_i \in M$) and there is a prior candidate function $N_j$ ($N_j \in N$) under which resource $R_j$ with the expression form of ‘-R’$ also exists, resource $R_j$ in $M_i$ can be obtained by extension of prior candidate function $N_j$.

**Proof:** Proof by contradiction is adopted. When resource expression form of current candidate function $M_i$ is ‘R’$, a prior candidate function $N_j$ ($N_j \in N$) exists; resource $R_j$ also exists under $N_j$ with the expression form of ‘-R’$, $M_i$ can be pruned. In line with description (1) in Definition 1, for resource $R_j$, ‘-1’ is under current candidate function in $M_i$. In accordance with the definitions of variant usage rate and low usage rate, resource $R_j$ under all functions of in $P$ may be ‘-1’ or ‘1’ under some functions. Since resource $R_j$ also exists under $N_j$ with the expression form of ‘-R’$, the bicluster extended currently may be a bicluster with low usage rate or a bicluster with variant usage rate. As the expression form of resource $R_j$ under current candidate function $M_i$ is ‘-1’, the bicluster $PN/M_i$ with variant usage rate or low usage rate can be gained through extension of $PN$. Thus, the bicluster with variant usage rate gained through extension of $PM$ can be obtained through extension of $PN/M_i$. Thus, $M_i$ can be pruned. This contradicts the assumption, so the original proof is established.

**Lemma 4:** assuming that $P$ is the bicluster with variant usage rate to be extended currently; $M$ is the candidate function set of $P$ and $N$ is the prior candidate function set of $P$. If the same prior candidate function $N_j(N_j \in N)$ exists for each resource $R_j$ in candidate function $M_i(M_i \in M)$, making each resource $R_j$ in candidate function $M_i$ meet the conditions in Lemma 1 or 2 or 3, candidate function $M_i$ can be pruned.

**Proof:** the process of proof can be gained through merging the processes of proof in Lemma 1, 2 and 3, so it is omitted here.

It can be seen from Lemma 4 that, the candidate function can only be pruned if all resources in the candidate function can be obtained by resource extension in the same prior candidate function; otherwise, this candidate function will be extended. If no successor or prior is its superset, it can be outputted. We will explain the algorithm mining process through an example. The data in the example are function-resource use relationship matrix shown in Table 1. Firstly, construct the weight graph among functions, as shown in Fig.1; then,
DoCluster algorithm deeply mines according to function extension.

(1) Firstly, the extension starts from $F_2F_2$, and all candidate functions of $F_1F_3$ are produced: $F_3(R_1R_2R_3)$ and $F_3(R_1R_2R_3)$. The resource conditions after the intersection is calculated are shown in the brackets. When candidate functions are produced, the resource set under candidate function $F_3$ of $F_1F_3$ can be gained only after the intersection of the weights of $F_1F_3$, $F_2F_3$ and $F_3F_3$ currently extended is calculated. Then, the candidate function $F_3(R_1R_2R_3)$ is produced through mining $F_1F_2F_3(R_1R_2R_3)$. Here, when producing the resource set under candidate function $F_3$, since the intersection of $F_1F_2F_3$ and $F_2F_3F_3$ has been worked out, it can be obtained through calculating the intersection of the weights of $F_1F_2F_3$, $F_2F_3F_3$ and $F_3F_3$, without the need of calculating the intersection of each edge. So, the maximal bicluster $F_1F_2F_3(R_1R_2R_3)$ can be gained through extending $F_2F_3$ deeply and preferentially. Then, prepare to extend $F_2F_3F_3(R_1R_2R_3)$. For $F_1F_2$, when all resources in current candidate function meet pruning conditions in Lemma 1, 2 or 3 for prior $F_3$. Therefore, $F_1F_2F_3$ can be pruned according to Lemma 4.

(2) Next, branch $F_1F_3$ is produced. All candidate functions of $F_1F_3(R_1R_2R_3)$ are $F_3(R_1R_2R_3)$ and $F_3(R_2R_3)$. For $F_2$, a prior candidate function $F_3(R_1R_2R_3)$ of $F_2F_3$ can be found. All resources of $F_3(R_1R_2R_3)$ are the subset of resources in $F_3(R_1R_2R_3)$, but resource $R_2$ does not meet pruning conditions (Lemma 3). So, $F_1F_3F_3$ can continue to be extended, but cannot be outputted. Then, the candidate function $F_3(R_2)$ is generated through extending $F_1F_3F_3(R_1R_2R_3)$. Since $F_3(R_2)$ does not meet pruning conditions, $F_1F_3F_3F_3$ can be outputted. When preparing to extend $F_1F_3F_3F_3$, a prior $F_3$ makes $F_3(R_2)$ satisfy the pruning conditions in Lemma 1, $F_1F_2F_3F_3$ should be pruned. Similarly, the branches of $F_2$, $F_3$ and $F_3$ can be mined respectively.

(3) When $F_2$ is mined, its candidate functions are $F_3(-R_1R_2R_3)$, $F_3(R_1R_2R_3)$ and $F_3(R_1R_2R_3)$, its prior candidate function is $F_3(R_1R_2R_3)$. As resources under $F_3F_3(R_1R_2R_3)$ do not satisfy pruning conditions, it is necessary to continue to extend $F_3F_3(R_1R_2R_3)$ whose candidate functions are $F_3(-R_1R_2R_3)$ and $F_3(R_1R_2R_3)$ and prior candidate function is $F_3(R_1R_2R_3)$. The candidate function $F_3(A_1R_2R_3)$ dissatisfies pruning conditions, so it is necessary to continue extending to generate $F_3F_3F_3(R_1R_2R_3)$. The candidate function is $F_3(R_1R_2R_3)$ and the prior candidate function is $F_3(R_1R_2R_3)$. Then, $F_1F_2F_3F_3(R_1R_2R_3)$ can continue to be generated and outputted. According to pruning conditions, $F_3F_3$ and $F_3F_3$ can be pruned.

(4) When extending to $F_3$, the candidate functions of $F_3$ are produced: $F_3(R_1R_2R_3)$ and $F_3(R_2R_3)$. Its prior candidate functions are $F_3(R_1R_2R_3)$ and $F_3(R_1R_2R_3)$. As resources in $F_3(R_1R_2R_3)$ are the subset in prior candidate function $F_3(R_1R_2R_3)$, $F_3F_3F_3(R_1R_2R_3)$ cannot be outputted. But $F_3F_3F_3(R_1R_2R_3)$ satisfies pruning conditions, so it is necessary to continue extending $F_3F_3F_3(-R_1R_2R_3)$ to produce the candidate function $F_3F_3F_3(R_1R_2R_3)$ and prior candidate functions: $F_3F_3F_3(R_1R_2R_3)$ and $F_3F_3F_3(R_1R_2R_3)$. At this moment, $F_3$ dissatisfies pruning conditions, so $F_3F_3F_3F_3F_3$ can be produced and outputted. For $F_3F_3F_3F_3F_3$, a prior $F_3F_3F_3F_3$ exists, making $F_3F_3F_3F_3$ satisfy pruning conditions, so $F_3F_3F_3F_3F_3$ is pruned.

(5) When extending $F_2F_3F_3F_3F_3$, a prior $F_3F_3F_3$ exists, making $F_3F_3F_3$ satisfy pruning conditions, so $F_3F_3F_3$ is pruned.

The above mining process is shown in Fig.3. The specific description of DoCluster algorithm is as follows:

Algorithm 1: DoCluster algorithm
Input: number threshold: $r_{min}$, function-resource matrix: $D$
Output: all biclusters with maximal variant usage rate or maximal low usage rate meeting the threshold
Initial value: sample weight graph: $G_0$ = Null, current bicluster to be extended $Q_0$ = Null, $S_0$ = Null and $S_0$ = Null.
Algorithm description: DoCluster($r_{min}, D, Q, S_0, S_0$)

(1) If $G_0$ is null, scan data set $D$ and construct its weighted graph. $S_0$ is the first sample in the weighted graph;

(2) For each sample $S_0$ connected with sample $S_0$

(3) If all resource linked lists in $S_0$ satisfy pruning conditions in Lemma 4, then

(4) Continue;

(5) Else

(6) For resource linked lists not satisfying pruning conditions, $Q_0Sample = Q_0Sample \cup S_0$;

$Q_0Resource = Q_0Resource \cup S_0$; Resource;

(7) DoCluster($r_{min}, D, Q_0, S_0, S_0$); next);

(8) Endif

(9) Endfor

(10) If $Q$ satisfies maximal definition, then

(11) Output $Q$

(12) Endif;

(13) $S_0 = S_0$ next;

(14) Return
Figure 3. Example mining process of DoCluster algorithm

Figure 3. The comparison of performance periods of the above three algorithms under each data set when the number of functions is 20: (a) 200 resources; (b) 500 resources; (c) 800 resources

Figure 4. The comparison of performance periods of the above three algorithms under each data set when the number of functions is 35: (a) 200 resources; (b) 500 resources; (c) 800 resources

Figure 5. The comparison of performance periods of the above three algorithms under each data set when the number of functions is 50: (a) 200 resources; (b) 500 resources

IV. EXPERIMENTAL RESULT AND ANALYSIS

In this section, we will make an experimental comparison on the mining efficiency and result of the algorithm above and existing algorithms. The hardware environment of the experiment is desktop computer: Intel(R) Core(TM)2 Duo 2.53GHz CPU and 4G memory; the software environment is Microsoft Windows 7 SP1 operating system; the algorithm programming and operating environment is Microsoft Visual C++ 6.0 SP6. Experimental data used in this paper are simulation data. To fully test the performance of the algorithm, we produce five data sets randomly, each of which contains 50 functions and 800 resources. Table 6 describes proportions of 1, 0 and -1 in each row in each data set.
In this section, the comparison will be made on the mining efficiency of DoCluster algorithm. FDCluster_One algorithm and MMCCluster algorithm. FDCluster_One algorithm adopts prior detection method described in literature [18]: mine maximal bicluster from discretized matrix data without candidate maintenance. The mining process of MMCCluster algorithm and FDCluster_One algorithm is basically the same. The difference is that during design of pruning strategy, MMCCluster algorithm first judges whether the gene set of current potential samples is the subset of a prior candidate sample set, while FDCluster_One algorithm first calculates the intersection and then judges prior samples.

The mining efficiency of the above three algorithms is compared as follows. To fully compare the scalability of algorithms, we produce multiple groups of data sets with different numbers of resources and functions in allusion to five data sets in Table 6. The selection of resources and functions are based on the order of resources and functions in data set. Figures 4(a)-4(c) provide the comparison of performance periods of the above three algorithms under each data set when the number of functions is 20 and the number of resources is 200, 500 and 800 respectively. It can be seen from these figures that the mining time of each algorithm increases progressively with the increase of the proportion of ‘-1’ in the data set. This is because the biclusters with variant usage rate and low usage rate do not restrain the number of ‘-1’. Thus, as the number of ‘-1’ increases in the data set, the scale of the bicluster mined will increase continuously, thus increasing mining complexity of each algorithm. It thus can be seen, for mining of function-resource matrix, the number proportion of ‘-1’ in the data set. This is because the biclusters with variant usage rate or low usage rate do not restrain the number of ‘-1’. Thus, as the number of ‘-1’ increases in the data set, the scale of the bicluster mined will increase continuously, thus increasing mining complexity of each algorithm. It thus can be seen, for mining of function-resource matrix, the number proportion of ‘-1’ in the data set directly influences the complexity of the algorithm. But, when the proportion of ‘-1’ is certain, as the proportion of ‘1’ increases in the data set, the complexity of the algorithm also increases. For data sets D1 and D2, the three algorithms can complete mining within 0.5s. The efficiency superiority of DoCluster algorithm is not obvious. However, as the proportion of ‘-1’ in the data set increases, in data sets D1, D2 and D3, the pruning strategy of DoCluster algorithm displays efficiency superiority.

To further test and verify the scalability of algorithms, figures 5(a)-5(c) provide the comparison of performance periods of the above three algorithms under each data set when the number of functions is 35 and the number of resources is 200, 500 and 800 respectively; figures 6(a)-6(c) provide the comparison of performance periods of the above three algorithms under each data set when the number of functions is 50 and the number of resources is 200 and 500, respectively. It can be seen from these figures that the mining efficiency of the three algorithms declines significantly compared with Fig.4 with the increase in the number of functions. This is because the three algorithms adopt row extension for mining. As the number of samples in the data set increases, mining depth and complexity of the algorithms increase. Meanwhile, the number of prior candidate samples for pruning judgment will also increase, thus increasing pruning complexity. In most data sets shown in Fig.5 and 6, the mining efficiency of DoCluster algorithm is the highest. However, in the data sets with large proportion of ‘1’, DoCluster algorithm fails to show the advantage of mining efficiency. This may be because multiple biclusters including ‘1’ can exist simultaneously in the biclusters mined by MMCCluster algorithm and FDCluster_One algorithm, while at most one ‘1’ can be included in a bicluster mined by DoCluster algorithm due to the restraint of the bicluster with variant usage rate. So, the number of biclusters mined by DoCluster algorithm is greater than the above two algorithms, thus including the pruning efficiency of the algorithm.

V. CONCLUSION

This paper proposed an efficient algorithm - DoCluster algorithm which can effectively mine all biclusters with maximal variant usage rate and low usage rate from the discrete function-resource matrix. First, this algorithm constructs a sample weighted graph which includes all resource collections between both samples that satisfy the definition of variant usage rate or low usage rate; then, all biclusters with maximal variant usage rate and low usage rate meeting the definition are mined with the mining method of using sample-growth and depth-first method in the constructed weighted graph. To improve the mining efficiency of the algorithm, DoCluster algorithm uses several pruning strategies to ensure mining maximal bicluster without candidate maintenance. However, original data information will be lost if the mining is conducted in discrete data. Our next research direction is to mine biclusters with variant usage rate and low usage rate in real function-resource matrix.

ACKNOWLEDGMENT

This paper is supported by National Key Basic Research Program of China under Grant No. 2014CB744900.

REFERENCES


Lihua Zhang is a doctoral student at the school of computer science and engineering at the northwestern polytechnical university, Xi'an China. She completed her master degree from northwestern polytechnical university in 2008. Her current research interests are PHM, avionics, data mining and safety. Since 2013, she has been studying at science and technology on avionics integration laboratory.

Miao Wang is an engineer at science and technology on avionics integration laboratory. He completed his doctor and master degree from northwestern polytechnical university in 2013 and 2018, respectively. He is a member of China computer federation. His research interests mainly include data mining, PHM, avionics and safety.

Zhengjun Zhai is a professor at the school of computer science and engineering at the northwestern polytechnical university, Xi'an China. He is vice chairman of NPU youth association for science and technology, distinguished expert of aerospace electrical & electronics and weapon system Standardization technology committee, distinguished expert of AAMRI and premium member of china computer federation. His research interests include experiment and testing systems Integration, remote maintenance and fault diagnosis and virtual visualization.

Guoqing Wang is a professor and a supervisor of Ph.D. student in Northwestern Polytechnical University. He was born in 1956 and received his M.S. and Ph.D. degrees in computer science and technology from the Northwestern Polytechnical University in 1984 and 1991 respectively. He is the institute director of China aeronautical radio electronics research institute, and the director of science and technology on avionics integration laboratory. He has long been engaged in the related technical research of avionic system integration, distributed parallel processing, high reliable fault-tolerant system, network and bus system etc. He serves as the vice director of national serve environment computer academy.