Mining Frequent Closed Patterns using Sample-growth in Resource Effectiveness Data

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

***Abstract*—As the occurrence of failure of electronic resources is sudden, real-time record analysis on the effectiveness of all resources in the system can discover abnormal resources earlier and start using backup resources or restructure resources in time, thus managing abnormal situations and finally realizing health management of the system. This paper proposed an algorithm: *MFPattern*, for mining frequent closed resource patterns in resource effectiveness matrix. In order to improve the efficiency, *MFPattern* algorithm uses sample-growth method and effective pruning strategies to guarantee mining all frequent closed patterns without candidate maintenance. Different from the traditional frequent closed pattern, *MFPattern* algorithm can mine resource combination patterns with all resources very effectively during work, those with simultaneous failure of resources and combination patterns in which some resources are very effective while some others have failure. The experimental result shows that our algorithm is more effective than existing algorithms.**

***Index Terms*—frequent pattern, closed, resource**

I. Introduction

Since resources are the physical support of system, the effectiveness of resources will directly influence the effectiveness of the system. Due to the influence of environmental factors and performance degradation of materials, electronic resources in the system will inevitably experience the process of capability loss. The influence of external stress in the process of system operation might cause acceleration of the degradation process of electronic resources and even damage and failure in extreme situations. Due to the aggravation of degradation, if not found in time, resource fault might finally cause system failure. Therefore, research on resource effectiveness is the footstone of prognostics and health management [1] of the system. As the occurrence of failure of electronic resources is sudden, real-time record analysis on the effectiveness of all resources in the system can discover abnormal resources earlier and start using backup resources or restructure resources in time, thus managing abnormal situations and finally realizing health management of the system.

Due to the great number of resources in system and the huge size of effectiveness value matrix of resources sampled in a period of time, how to mine the required resource pattern from these data efficiently is vital for health management of the system. The widely used data mining technology can mine various required knowledge from a lot of complex data. Frequent pattern mining [2-10] is an important branch of data mining technology. It can mine patterns meeting some rules and occurring frequently from huge data. However, this method has a disadvantage, i.e. existence of excessive redundancy modes. Literatures [11-14] reduce redundant patterns for mining frequent closed pattern. The important feature of frequent closed pattern algorithm is reserving and detecting strategies with candidate item. However, when there are a great number of item-sets in data set, it will produce a lot of candidate item-sets and thus cause the breakdown of internal memory. Therefore, *BIDE*[15] algorithm proposed by Han et al. uses backward checking method to avoid frequent item reservation, thus it can improve the mining efficiency. However, when the data set is sparse, the method of backward checking has a low pruning efficiency, thus increasing the complexity of the algorithm. Then *WIBE*[16] algorithm proposed by Wang et al. judges frequent closed pattern based on item-set weight information detection. Specifically, all items in the current candidate item-set have the same weight, this pattern is the subset of a frequent closed gene pattern that has been output and can be pruned. However, if the data set is large or sparse, this algorithm needs to store a lot of weight information in the internal memory, which will influence its efficiency and cause the breakdown of internal memory. Pan et al. [17,18] proposed a sample enumeration method of exploring enumeration space through recursive establishment and transposition of project database. Due to the lack of effective pruning strategy, the mining efficiency of this algorithm will be influenced when the number of samples is great.

To discover abnormity or failure in time, there should not be excessive sampling sites when the effectiveness value of resources is collected. There are two reasons as follows: (1) excessive sampling sites will influence the mining efficiency of the algorithm, thus causing the failure to report abnormal situations of resources in time;

doi:10.4304/jcp.9.5.1150-1158

(2) excessive sampling sites will influence the real-time property of the system, thus causing the failure to ensure normal operation of the system and finally influencing the health of the whole system. Based on the analysis above, this paper proposes an algorithm: *MFPattern*, for mining frequent closed resource pattern in resource effectiveness matrix using sample-growth method. In order to improve the mining efficiency, this algorithm uses effective pruning strategies to ensure the mining of all frequent closed patterns without candidate maintenance. *MFPattern* algorithm can mine the following three patterns: (1) resource combination pattern with very effective resources during system operation; (2) resource combination pattern with simultaneous failure of resources during system operation; (3) resource combination pattern in which some other resources will inevitably have failure during efficient operation of some resources.

II. Problem Description

Resource effectiveness matrix is defined as a two-dimensional real matrix $D = R \times S$, where row collection R represents the resource names set and column collection S refers to a set of different sampling sites. Element D_{ij} of matrix D is a real-valued number which refers to the effective value (e.g. BIT value) of resource i under sampling site j. $|R|$ is the number of resources in data set D and $|S|$ is the number of sampling sites in data set D. For the convenience of mining, the original effective value in resource effectiveness matrix is generally discretized into 1, -1 and 0, where 1 represents resource health, 0 represents sub-health and -1 represents resource failure, as shown in table 1.

Discrete resource effectiveness matrix

	S_0	S_1	S_2	S_3	S_4
R_1	1	-1	1	-1	1
R_2	1	-1	1	-1	-1
R_3	0	1	-1	1	1
R_4	1	1	0	1	1
R_5	-1	-1	0	-1	1

The relationship between two resources R_1 and R_2 can be defined as follows: (1) if R_1 and R_2 are both very effective (the value is 1), they have effective positive correlation, expressed as R_1R_2. The support is $sup(R_1R_2) = \frac{|R_1R_2|}{|S|}$, where $|R_1R_2|$ is the number of sampling sites when R_1 and R_2 are both very effective; (2) if R_1 and R_2 both have failure (the value is -1), they have failure positive correlation, expressed as $-R_1$-R_2. The support is $sup(-R_1 - R_2) = \frac{|-R_1 - R_2|}{|S|}$, where $|-R_1\text{-}R_2|$ is the number of sampling sites when R_1 and R_2 are both failure; (3) if R_1 is very effective while R_2 has failure, they have effective negative correlation, expressed as R_1-R_2; if R_2 is very effective while R_1 has failure, they also have effective negative correlation, expressed as $-R_1$-R_2; when R_1 and R_2 have effective negative correlation, the support is $sup(R_1 - R_2) = \frac{|R_1 - R_2|}{|S|} + \frac{|-R_1R_2|}{|S|}$, where $|R_1\text{-}R_2|$ is the number of sampling sites when R_1 is very effective and R_2 is failure, and $|-R_1R_2|$ is the number of sampling sites when R_1 is failure and R_2 is very effective.

Among three relationships above, the first one mines resource combination patterns in which resources are all very effective, i.e. there is a high efficiency when these resources work together; the second one mines resource combination patterns in which resources have failure together; in such patterns, potential reasons of failure can be found and decisions can be made for the use of backup resources; the third one mines resource patterns when some resources are very effective while some other resources have failure, in which potential unstable resources can be found and replaced earlier.

The mining purpose of *MFPattern* algorithm is to mine all frequent closed resource patterns in resource effectiveness data set, i.e. mine resource patterns without superset and with the same support which satisfies the threshold of support. To improve the mining efficiency of the algorithm, *MFPattern* algorithm mines frequent closed patterns using sample-growth without candidate maintenance. The mining process of this algorithm will be introduced in the next section.

III. The MFPattern Algorithm

Most traditional methods for frequent closed pattern mining are based on row extension. There are two reasons as follows: (1) with row extension, apriori principle can be used to prune the current extended resource that not satisfying the support threshold in real time; (2) according to the definition of frequent closed pattern: if there is no superset with the same support as the current extended frequent pattern, the current extended frequent pattern is frequent closed pattern; it is more convenient to make closed judgment through mining with row extension; if there is no candidate item-set with the same support in all candidate item-sets of the current extended frequent pattern, the current frequent pattern must be frequent closed pattern.

However, there are some disadvantages of using row extension as follows: (1) resources might have three relations simultaneously. If the method of row extension is used for mining, the number of resource patterns produced will present 3^n increase in extreme situations, thus row extension increases the complexity of mining; (2) if there are many resources and a few sampling sites, the use of row extension will increase the number of iterations of the algorithm, thus influencing the time efficiency and space efficiency of the algorithm; (3) as resource sampling is dynamic, with the use of row extension, the support of resources will change when the column of original data increases, thus making the design of incremental frequent closed pattern mining algorithm

relatively complex. Therefore, *MFPattern* algorithm uses the method of sample-growth for mining.

Before introducing how *MFPattern* algorithm mines frequent closed resource patterns without candidate maintenance in detail, we will first analyze mining strategies of existing frequent closed pattern mining algorithms briefly. Generally speaking, there are three strategies for mining frequent closed patterns: (1) it mines all frequent patterns firstly, and then outputs frequent closed patterns meeting the condition according to the definition of frequent closed pattern. This mining method is simple and visual, but has a low efficiency; (2) it mines frequent closed patterns with the method of real-time monitoring, i.e. store frequent closed patterns produced currently in the internal memory and compare each new pattern produced with frequent closed patterns stored in memory. If any frequent closed pattern in memory has the same support with the current extended pattern and is also the superset of the current extended pattern, then it stops the current pattern extension; if it is a new frequent closed pattern, update frequent closed patterns stored in the internal memory till finishing the mining of all frequent closed patterns. Such method can make pruning judgment for the current extended pattern in real time. However, as it is required to store all frequent closed patterns produced in the internal memory, it wastes the storage space. Meanwhile, each new frequent pattern produced is compared with frequent closed patterns in the memory, thus it reduces the mining efficiency; (3) it makes pruning judgment for the current extended frequent pattern with the method of backward checking. If the current candidate pattern is the subset of a frequent closed pattern that has been mined, the current extended pattern must have a prior candidate resource (which is candidate resource that has been extended) satisfying the definition of frequent closed pattern for pruning judgment. This method can avoid storing frequent closed patterns in the internal memory for pruning. However, if original data are sparse, the success rate of pruning will decrease and this backward checking method is time-consuming.

As resources in the system are effective in most cases, health management will be conducted in the case of failure. Therefore, the resource effectiveness matrix used in this paper is dense. *MFPattern* will mine frequent closed resource patterns with the method of sample-growth and backward checking. The mining process of this algorithm can be divided into two steps: first, construct a sample weighted graph; then, mine all frequent closed resource patterns with the method of sample-growth.

A. Construct Undirected Sample Relational Weighted Graph

The method of mining patterns with sample weighted graph was first used in the mining of bicluster in *MicroCluster* algorithm [19]. Then, Wang et al. [20,21] used sample weighted graph to mine bicluster and fault-tolerant bicluster. *MFPattern* algorithm proposed in this paper will use undirected sample relational weighted graph (sample weighted graph for short, the same below) to mine frequent closed resource patterns.

Definition 1. Sample weighted graph can be expressed as $G = \{E, V, W\}$. Each vertex in vertex set V in the weighted graph represents a sampling site; if an edge exists between a pair of vertices, it means that there are resources with effectiveness relationship under two sampling sites represented by this pair of vertices. The collection of edge is expressed with E; the weight on each edge is resource collection with effectiveness relationship under two sampling sites connected with this edge. The weight set is expressed with W.

Fig.1 shows the sample weighted graph corresponding to table 1. Weight on each edge is denoted as the set of effective resources under two sampling sites connected with this edge. The effective relationship among resources is obtained through the third relationship among resources defined in section two of this paper.

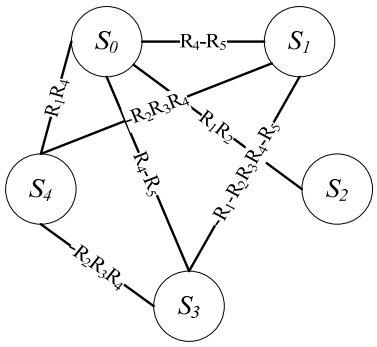

Figure 1. Undirected sample relational weighted graph corresponding to table 1

B. Mine Frequent Closed Resource Pattern

This section will introduce how *MFPattern* algorithm mines frequent closed resource patterns in the sample weighted graph. For the convenience of designing effective pruning strategies, *MFPattern* algorithm adopts the depth-first principle. It is assumed that frequent closed resource patterns are mined from the sample weighted graph in Fig.1. Firstly, *MFPattern* algorithm extends the edge connected with S_0 node, i.e. S_0->S_0S_1->$S_0S_1S_2$; then, it extends S_1 after the extension of S_0 branch. The support of resource pattern under sample $S_0S_1S_2S_3S_4$ is 1; that under samples $S_0S_1S_2S_3$, $S_0S_1S_2S_4$, $S_0S_2S_3S_4$ and $S_1S_2S_3S_4$ is 0.8; that under samples $S_0S_1S_2$, $S_0S_1S_3$, ..., and $S_2S_3S_4$ is 0.6. It can be seen from the extension method above that the method based on sample-growth has a different process from the mining of frequent closed patterns with resource extension.

When using resource extension to mine, it produces the support of two item-sets and then extends to three and four item-sets, etc. According to Apriori principle, the support decreases with the increase of the number of resources in item-set. The size of frequent patterns (the number of resources in the pattern) mined under the same sample branch with the method of sample-growth arranges from large to small and the support of pattern increases progressively. As there are only two samples during the extension, with the increase of the number of samples, the number of resources satisfying the definition together under the sample must decrease. However, the

support of resource pattern increases (because the number of samples increases). Therefore, when frequent closed resource patterns are mined based on resource extension, extension can terminate as long as the support of resource does not meet the threshold value. However, when patterns are mined based on sample-growth, it is necessary to continue the extension of sample set so long as the weight under the sample set extended is not null or not more than the minimum number of resources defined by users.

Definition 2. P is the current extended resource pattern. If all samples in S_i and P have a link in the sample weighted graph and the number of intersections (expressed as S_i.*Resources*) of weights of all join edges satisfies the number defined by users, S_i is the candidate sample of P.

Definition 3. P is the current extended resource pattern. If S_i and S_j are candidate samples of P, but S_i has been extended by P and S_j has not been extended by P yet. For S_j, S_i is prior candidate sample of P.

It can be known from the descriptions of the definition above that, a candidate sample becomes prior candidate sample if it has been extended. According to the principle of mining frequent closed patterns with backward checking, if the resource under the current extended sample set is the subset of resources under a prior candidate sample or candidate sample, the resource pattern under the current extended sample must not be frequent closed resource pattern. For example, assuming that the order of sample extension is from S_0 to S_4 based on the name and the current extended sample set is S_0S_1, S_3 and S_4 are its candidate samples and S_2 is its prior candidate sample (i.e. mining of $S_0S_1S_2$ branch has been completed). If the current sample to be extended is S_3, for $S_0S_1S_3$, S_2 is its prior candidate sample. It means that $S_0S_1S_2S_3$ must have been extended before $S_0S_1S_3$ is extended. If the resource pattern under $S_0S_1S_3$ is a subset of that under $S_0S_1S_2$, the resource pattern that can be obtained by extension of $S_0S_1S_3$ can definitely be obtained by extension of $S_0S_1S_2$ as well as $S_0S_1S_2S_3$ (because the resource pattern under $S_0S_1S_2S_3$ is the same as that under $S_0S_1S_3$ at this time); as the support of $S_0S_1S_2S_3$ is higher than that of $S_0S_1S_3$, $S_0S_1S_3$ can be pruned according to the definition of frequent closed pattern.

Lemma 1. Assume that P is the current extended resource pattern, M is the candidate sample set of P and N is the prior candidate sample set of P, if a prior candidate sample $N_j(N_j \in N)$ making PM_i. *Resource* a subset of PN_j.*Resource* exists for candidate sample $M_i(M_i \in M)$, PM_i. *Resource* should be pruned.

Proof. Proof by contradiction. Assuming that the resource set of the current candidate sample M_i is not a subset of resource collection of a prior candidate sample N_j before it, M_i can be pruned. It can be known from the assumption that resources not belonging to PN_j.*Resource* exists in PM_i.*Resource*. As frequent closed pattern mining uses sample depth-first extension method and N_j is extended earlier than M_i, there might be another sample S_m making PM_iS_m.*Resource* is not equal to $PM_iN_jS_m$.*Resource*. Therefore, M_i cannot be pruned, which is contradictory with the assumption. Thus, the original evidence is true.

Candidate samples meeting theorem 1 should be directly pruned and those not meeting pruning conditions should continue extension. However, whether the current extended pattern is output, should be judged according to the following output strategy (which actually meets the definition of frequent closed pattern):

Output strategy. Assuming that P is the current frequent closed pattern to be extended, M is candidate sample set of P and N is prior candidate sample set of P, if there is no candidate sample M_i making P.*Resource* a subset of PM_i.*Resource*for all candidate samples $M_i(M_i \in M)$ of P, P.*Resource* can be output.

Based on the analysis above, *MFPattern* algorithm can directly mine frequent closed resource pattern with the method of sample-growth without storing candidate frequent pattern in memory. Fig.2 illustrates the mining process of *MFPattern* algorithm. Example data is shown in table 1 and the threshold of support is 0.4 and the minimum number of resources in pattern is 2.

Algorithm 1: *MFPattern* algorithm

Input: threshold of support: r_{min}; resource effectiveness data: D

Output: all frequent closed resource patterns satisfying the threshold value

Initial value: sample weighted graph: G =Null; current pattern to be extended Q =Null，S_i=Null，S_j=Null

Algorithm description: MFPattern(r_{min}, D, Q, S_i, S_j)

(1) If G is null, scan data set D and make its weighted graph. S_i is the first sample in the weighted graph;

(2) For each sample S_j connected with sample S_i,

(3) If all resources linked lists in S_j satisfy pruning conditions,

(4) Continue;

(5) Else

(6) For resource linked lists not satisfying pruning conditions, $Q.Sample = Q.Sample \cup S_j$; $Q.Resource = Q.Resource \cap S_iS_j.Resource$;

(7) MFPattern(r_{min}, D, Q, S_i, S_j->next);

(8) end if

(9) end for

(10) if Q satisfies output conditions, then

(11) Output Q;

(12) end if;

(13) $S_i = S_i$->next;

(14) return

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 internal 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 is simulation data. To fully test the performance of the

algorithm, we generate six data sets randomly. Each data set contains 35 sampling sites and 800 resources. Table 2 describes the proportion of 1, 0 and -1 in each row in each data set.

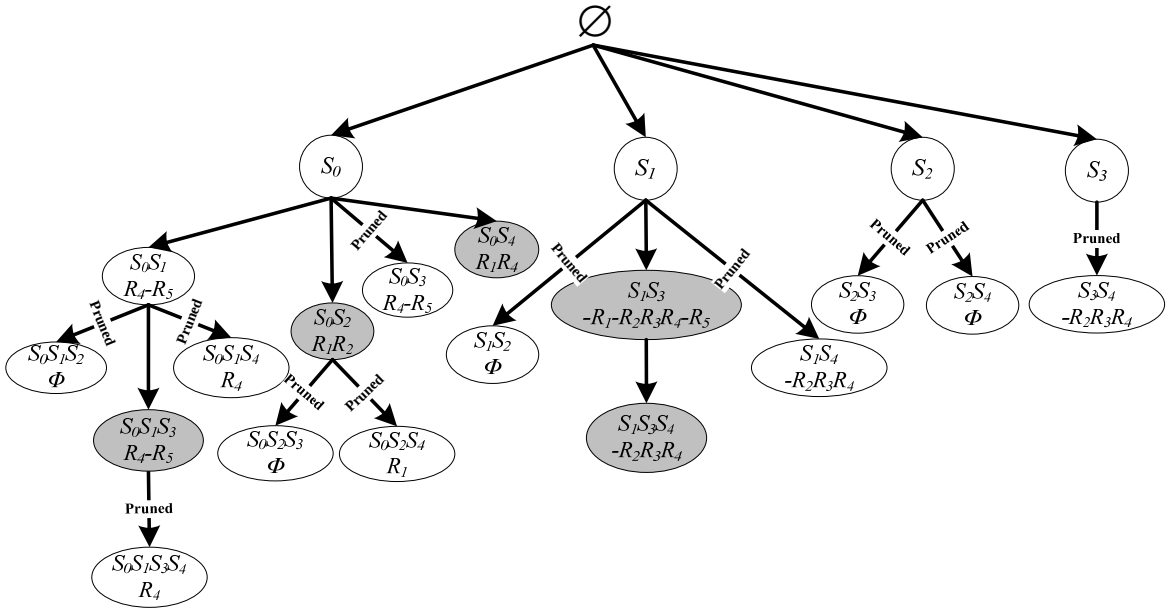

Figure 2. Example mining process of MFPattern algorithm

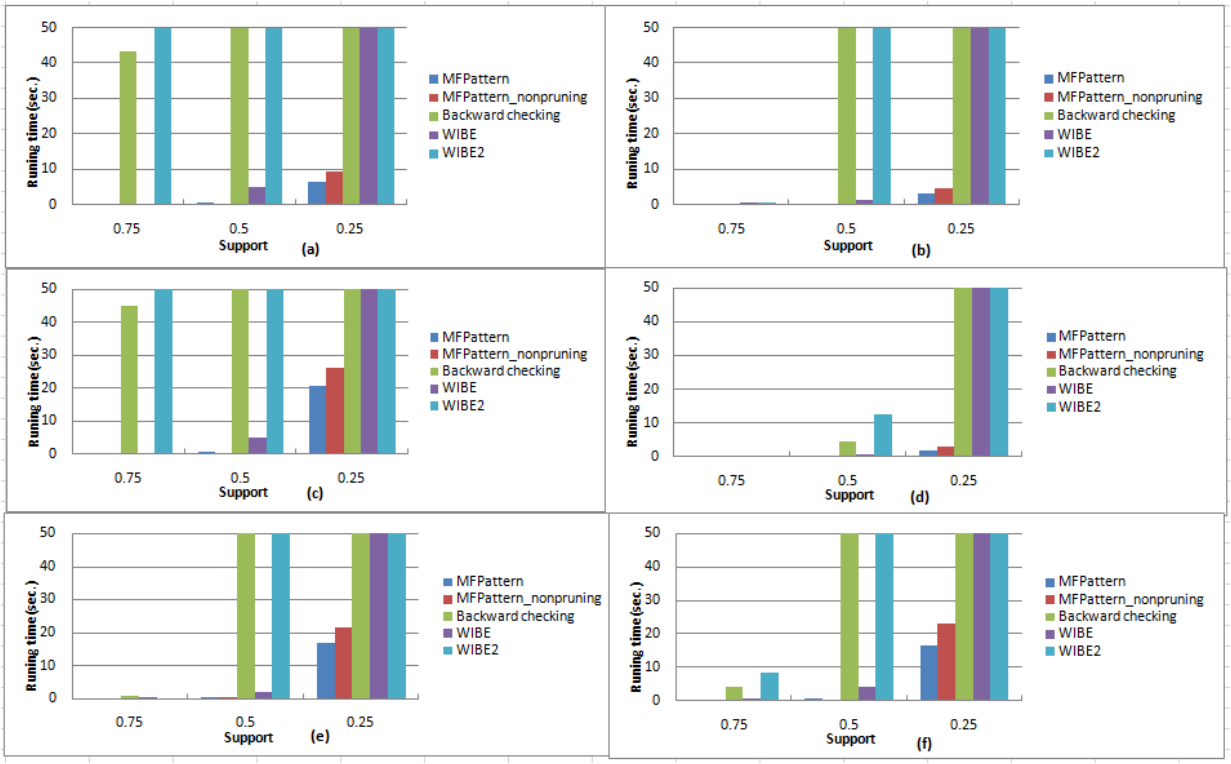

Figure 3. Comparison of operating time of five algorithms under different data sets with 200 resources and 20 sampling sites: (a) D_1; (b) D_2; (c) D_3; (d) D_4; (e) D_5; (f) D_6

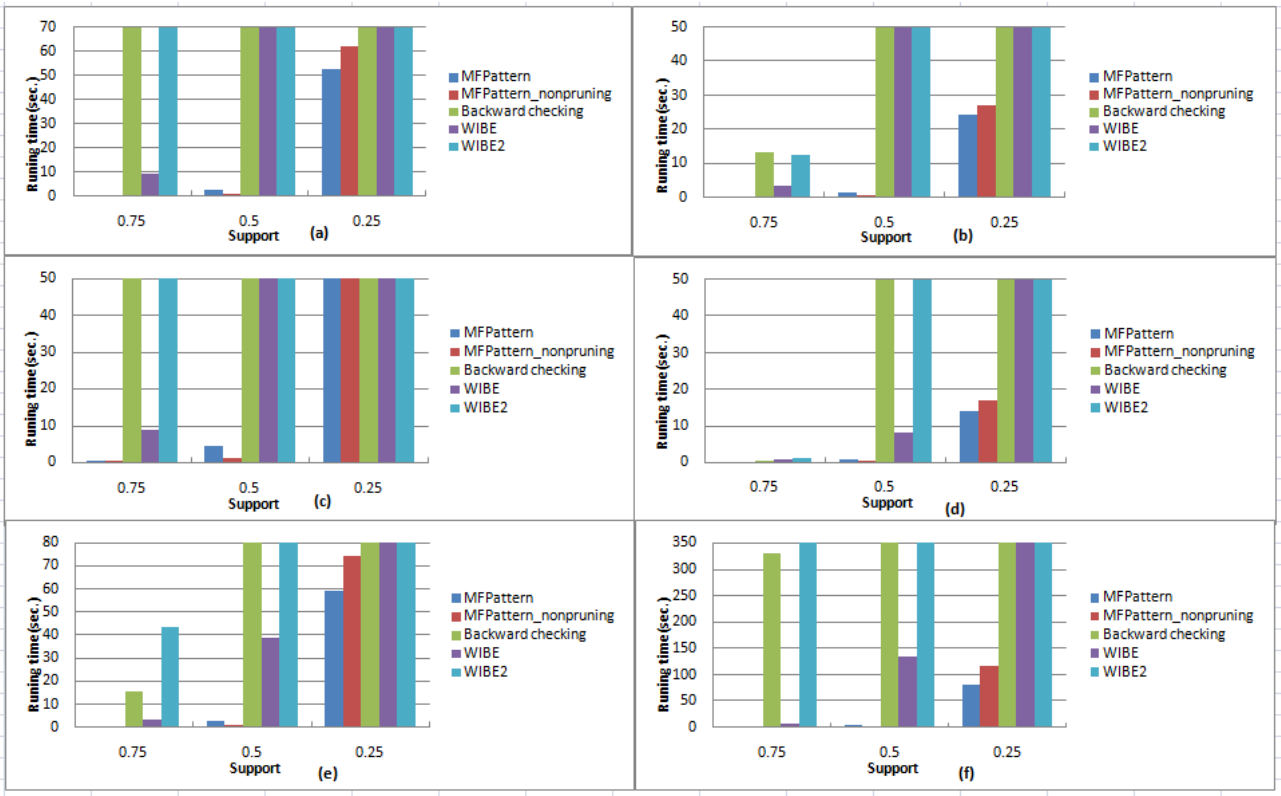

Figure 4. Comparison of operating time of five algorithms under different data sets with 500 resources and 20 sampling sites: (a) D_1; (b) D_2; (c) D_3; (d) D_4; (e) D_5; (f) D_6

DISTRIBUTION OF NUMERICAL VALUE PROPORTION IN SIX DATA SETS

	S_0	S_1	S_2	S_3	S_4
R_1	1	-1	1	-1	1
R_2	1	-1	1	-1	-1
R_3	0	1	-1	1	1
R_4	1	1	0	1	1
R_5	-1	-1	0	-1	1

In this section, *MFPattern* algorithm will be compared with *MFPattern* algorithm without pruning (denoted as *MFPattern_nonpruning*), *Backward Checking* algorithm, *WIBE* algorithm and *WIBE2* algorithm. *MFPattern_nonpruning* algorithm uses the same mining method as *MFPattern* algorithm, i.e. mine frequent closed patterns with the method of sample-growth. Different from *MFPattern* algorithm, *MFPattern nonpruning* algorithm does not use pruning strategies described in Lemma 1, but mines with the method of full extension. *Backward Checking* algorithm uses backward checking method described in literature [15] and mines frequent closed patterns without storing frequent item-sets. *WIBE* algorithm uses pruning strategies described in literature [16] and can mine frequent closed patterns without candidate maintenance. *WIBE2* algorithm uses the same pruning strategies as *WIBE* algorithm to mine frequent closed patterns. Different from *WIBE* algorithm, *WIBE2* algorithm uses co-expression support for mining.

The mining efficiency of five algorithms above will be compared. To fully compare the extendibility of algorithms, we produce multiple groups of data sets with different numbers of resources and sampling sites in allusion to six data sets in Table 2. Resources and sampling sites are selected according to the order of resources and sampling sites in data sets. Figs 3(a)-3(f) provide the comparison of operating time of five algorithms above under different data sets with 200 resources and 20 sampling sites. It can be seen from these figures that *MFPattern* and *MFPattern_nonpruning* algorithms can complete the mining process within 1 second and have more mining efficiency than other algorithms when the support is 0.75 and 0.5 under each data set. As *Backward Checking* algorithm judges frequent closed patterns with the method of backward checking, when the data set is dense (the proportion of 0 is low), it is necessary to frequently use backward checking strategy. Thus, the mining efficiency is low. Therefore, even when the support is 0.75, *Backward Checking* algorithm cannot complete the mining process in limited memory space under dense D_1 and D_3 data sets. When the support is 0.5, *Backward Checking* algorithm can only complete the mining process under data set D_4 with the lowest density of data. When the support is 0.25, only *MFPattern* and *MFPattern_nonpruning* algorithms can complete the mining process and other three algorithms cannot complete mining in limited memory space. As more frequent closed patterns can be produced when the support is low, pruning strategies used in

MFPattern algorithm can reduce the mining space of the algorithm. Thus, it has a higher mining efficiency than *MFPattern_nonpruning* algorithm. It is thus clear that the mining of frequent closed patterns with sample-growth is highly efficient and meanwhile pruning strategies used in *MFPattern* algorithm can improve the mining efficiency of the algorithm.

To further verify the extendibility of algorithms, Figs.4(a)-4(f) provide the comparison of operating time of five algorithms above under different data sets with 500 resources and 20 sampling sites. It can be seen that, similar to descriptions in Fig.3, *MFPattern* and *MFPattern_nonpruning* algorithms have more efficiency than other algorithms under different supports in each data set. When the support is 0.25, pruning strategies used in *MFPattern* algorithm have more obvious advantages than other supports. Figs.5(a)-5(f) provide the comparison of operating time of *MFPattern* algorithm and other three algorithms under different data sets with 500 resources and 35 sampling sites. It can be seen that *MFPattern* algorithm has the fastest mining process under most data sets and supports. However, compared to the operating time in Fig.3 and Fig.4, the operating time of *MFPattern* algorithm increases with the increase of the number of sampling sites. When the support is 0.55, as shown in Fig.5(c) and Fig.5(3), *MFPattern* algorithm has a lower mining efficiency than *WIBE* algorithm. As both data sets are dense, *MFPattern* algorithm needs frequent pruning judgment, thus influencing the mining efficiency. When the support is 0.4, *Backward Checking* and *WIBE2* algorithms cannot complete the mining process under all data sets in limited memory space. *MFPattern* algorithm can complete the mining process under sparse data sets D_2 and D_4.

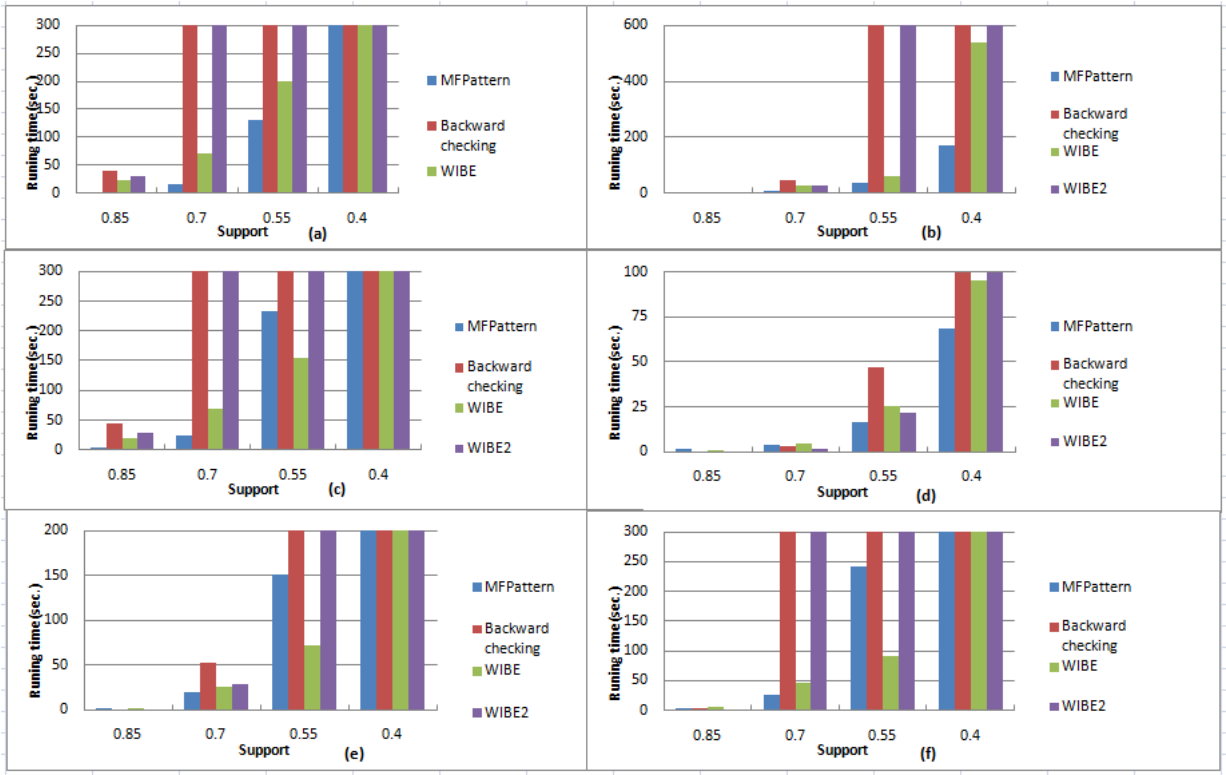

Figure 5. Comparison of operating time of five algorithms under different data sets with 800 resources and 35 sampling sites: (a) D_1; (b) D_2; (c) D_3; (d) D_4; (e) D_5; (f) D_6

V. CONCLUSION

This paper proposed an algorithm mining frequent closed resource patterns from data effectiveness matrix with the method of sample-growth: *MFPattern*, which uses effective pruning strategies to guarantee the mining of all frequent closed patterns without maintaining candidate item-sets. Different from the traditional frequent closed pattern, *MFPattern* algorithm can mine resource combination patterns with all resources effectively during work, those with simultaneous failure of resources and combination patterns in which some resources are very effective while some other resources have failure. The experimental result shows that this algorithm is more efficient than existing mining methods of frequent closed pattern. However, mining on discrete data will cause the loss of original data information. Our next research direction is to mine frequent closed patterns related to resource health from real resource effectiveness data.

ACKNOWLEDGMENT

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

REFERENCES

[1] Michael Pecht,et al.. A prognostics and health management roadmap for information and electronics-rich systems. Microelectronics Reliability, 2010:317–323.

[2] J. Han, J. Pei, B. Mortazavi-Asl, Q. Chen, U. Dayal, and M.-C. Hsu, FreeSpan: Frequent Pattern-Projected Sequential Pattern Mining, Proc. 2000 ACM SIGKDD Int'l Conf. Knowledge Discovery in Databases (KDD '00), pp. 355-359, Aug. 2000.

[3] Pei Jian, Han Jiawei. Mining Sequential Patterns by Pattern-growth: The PrefixSpan Approach. IEEE Transactions on Knowledge and Data Engineering, 2004, 6(10): 1-17.

[4] Nohuddin P N E, Coenen F, Christley R, et al. Finding "interesting" trends in social networks using frequent pattern mining and self organizing maps. Knowledge-Based Systems, 2012, 29: 104-113.

[5] Mishra S, Mishra D, Satapathy S K. Fuzzy Frequent Pattern Mining from Gene Expression Data using Dynamic Multi-Swarm Particle Swarm Optimization. Procedia Technology, 2012, 4: 797-801.

[6] Pyun G, Yun U. A Frequent Pattern Mining Technique for Ranking Webpages Based on Topics. Multimedia and Ubiquitous Engineering. Springer Netherlands, 2013: 121-128.

[7] Glatz E, Mavromatidis S, Ager B, et al. Visualizing big network traffic data using frequent pattern mining and hypergraphs. Computing, 2013: 1-12.

[8] Yun U, Lee G, Kim S J. Analyzing Efficient Algorithms of Frequent Pattern Mining. IT Convergence and Security 2012. Springer Netherlands, 2013: 937-945.

[9] Xufei Zheng, Yanhui Zhou, Yonghui Fang. The Dual Negative Selection Algorithm Based on Pattern Recognition Receptor Theory and Its Application in Two-class Data Classification. Journal of computers, Vol 8, No 8, 2013, p:1951-1959.

[10] Haichao Luo, Xiaomin Li, Shujing Zheng, Mei Li, Lili Song. Study on Synthesis Evaluation of Intensive Land Use and Growth Pattern Transformation of Towns. Journal of computers, Vol 7, No 8, 2012, p:1959-1966.

[11] Pasquier N, Bastide Y, Taouil R, Lakhal L. Discovering frequent closed itemsets for association rules. In: Beeri C, et al, eds. Proc.of the 7th Int'l. Conf. on Database Theory. Jerusalem: Springer-Verlag, 1999. 398~416.

[12] Pei J, Han J, Mao R. CLOSET: An efficient algorithm for mining frequent closed itemsets. In: Gunopulos D, et al, eds. Proc. of the 2000 ACM SIGMOD Int'l. Workshop on Data Mining and Knowledge Discovery. Dallas: ACM Press, 2000. 21-30.

[13] Meng Han, Zhihai Wang, Jidong Yuan. Closed Sequential Pattern Mining in High Dimensional Sequences. Journal of Software, Vol 8, No 6, 2013, p:1368-1373.

[14] Qunhui Wu, Shilong Ma, Hao Wang. Extracting Feature Sequences in Software Vulnerabilities Based on Closed Sequential Pattern Mining. Journal of Software, Vol 8, No 8, 2013, p:1809-1817.

[15] Wang, J, Han, J. BIDE: Efficient Mining of Frequent Closed Sequences, Data Engineering, 2004. Proceedings. p: 79 – 90.

[16] Miao Wang, Xuequn Shang, Jingni Diao, Zhanhuai Li. WIBE: Mining Frequent Closed Patterns Without Candidate Maintenance in Microarray Dataset. DMIN 2010: 200-205.

[17] Cong, G., Tan, K., Tung, A. et al.: Mining Frequent Closed Patterns in Microarray Data. ICDM'04. IEEE Press, 2004, 363–366.

[18] Pan, F., Cong, G., Tung, K., Yang, J., Zaki, M.. Carpenter: Finding closed patterns in long biological datasets. In: Proc. ACM SIGKDD Intl. Conf. Knowledge Discovery and Data Mining (KDD), 2003, pp. 637–642.

[19] Lizhuang Zhao, Mohammed J. Zaki, MicroCluster: An Efficient Deterministic Biclustering Algorithm for Microarray Data, in IEEE Intelligent Systems, special issue on Data Mining for Bioinformatics, 2005, Vol. 20, No. 6, pp: 40-49.

[20] Miao Wang, Xuequn Shang, Miao Miao, Zhanhuai Li, Wenbin Liu. FTCluster: Efficient Mining Fault-Tolerant Biclusters in Microarray Dataset. Proceedings of ICDM 2011 workshop on Biological Data Mining and its Applications in Healthcare, p 1075-1082.

[21] Miao Wang, Xuequn Shang, Shaohua Zhang, Zhanhuai Li. FDCluster: Mining frequent closed discriminative bicluster without candidate maintenance in multiple microarray datasets. ICDM 2010 workshop on Biological Data Mining and its Applications in Healthcare, p 779-786.

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 degree and master degree from Northwestern Polytechnical University in 2013 and 2008, 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.