A Software Testing Optimization Method Based on Negative Association Analysis

Lin Wan[1], Qiuling Fan[1],Qinzhao Wang[2]

1Department of Information Engineering, Academy of Armored Force Engineering, Beijing, 100072, China

2Department of Control Engineering, Academy of Armored Force Engineering, Beijing, 100072, China

Keywords: Software defect; data matrix; negative association rules; software testing optimization.

Abstract. Association rules mining oriented to software defect data plays a significant role in exploring the relationship between software defect data. What's more, it can provide guidance for software developers and testers. What negative association rules describe is mutually exclusive relationship between the different projects, which has very important significance. By introducing vector space model and data matrixes, a novel software testing optimization method based on negative association analysis which is named GMNAR is proposed. And it is proved effective and efficient through experiments.

1.Introduction

Recently, software testing technology developed rapidly. Each testing institution has accumulated a lot of software defect data with high complexity by performing extensive testing work. Mining the negative association relationship of software defect data has very important significance. According to the association relationship between software defect data, software testers can determine which test should be deleted or which test should be strengthened. Then the test time can be reduced and test efficiency can be improved.

Currently, research methods on the positive association rules are relatively mature, but research methods on the negative association rules[1-3] are rare to some extent. Besides, these methods have the problem of frequently scanning databases and generating excessive candidate sets. In this thesis, a software testing optimization method based on negative association analysis is proposed, whose purpose is to solve the problem of immaturity of negative relationship analysis method, low efficiency and low reliability. What's more, the whole analysis process can be completed by scanning the database once. The analysis of negative association relationship can help testers filtering testing projects unrelated to defects and improve the efficiency of software testing.

2.Negative association rules extraction based on vector space model and data matrixes

In the paper, GMNAR method is presented by using the vector space model and matrix approach. The main idea is: vector space model of defect data can be established based on data discretization, so the similarity between different defect data is calculated, and then data matrix is generated through once database scanning, which can form the initial chromosome set of the genetic algorithm. Finally, negative association rules are extraction by rule evaluation strategy.

2.1Data discretization

Firstly, software defect data is discretized so that it can be converted to data that can be mined. In this thesis, rough set theory[4] is introduced. Decision system S represents transaction sets T. Combined with the characteristics of the negative association rules, S is defined as follows: S=<U,P,X,Y,V,f>. And the Parameters meaning of S is shown in Table 1.

Table 1: Parameter meaning of S

No.	Parameter name	Meaning
1	U	project to be analyzed
2	P	items in database
3	X	first components of negative association rule
4	Y	second components of negative association rule
5	V	$V = \bigcup_{P \in A} V_p$, V_p is the value of P
6	f	$f : U \times (X \bigcup Y) \to V$

Depending on V_p, different coding methods can be taken to discretize data in transaction set T.

i) If the value of V_p is limited and discrete, then these values are coded as "1", "2", ...;

ii) If the value of V_p is continuous or infinite, then equidistant division method is used to divide these values into a finite number of intervals and they were coded as "1", "2", ...;

iii) If the value of an item is empty or a transaction not including the item, then it is coded as "0."

2.2Creation of the Vector Space Model

When defect data are discretized, each of the defect data can be mapped to a vector represented by a characteristic number. Then all defect data can be mapped to a multi-dimensional matrix. This multi-dimensional matrix is the software defect data vector space model proposed in this thesis. Its math form is shown as:

$$D = \begin{pmatrix} d_{11} & d_{12} & d_{13} & d_{14} & d_{15} & d_{16} & d_{17} \\ d_{21} & d_{22} & d_{23} & d_{24} & d_{25} & d_{26} & d_{27} \\ d_{31} & d_{32} & d_{33} & d_{34} & d_{35} & d_{36} & d_{37} \\ \cdots & \cdots & \cdots & \cdots & \cdots & \cdots & \cdots \\ d_{i1} & d_{i2} & d_{i3} & d_{i4} & d_{i5} & d_{i6} & d_{i7} \end{pmatrix}$$

In the matrix, one line express one defect data, one column express one characteristic attribute. D_{ij} is the abstract code of every characteristic attribute.

2.3Defect data similarity detection

In this article, the similarity between different defect data is calculated by introducing fuzzy equivalent matrixes. It makes full use of the advantages of the matrix method, including simple and efficient, to improve the accuracy and efficiency of the calculation.

2.3.1Data normalization

In order to avoid bad effect caused by great difference of subsequent data on the calculation accuracy and speed up the convergence program, software defect data vector space model D must be normalized first. It means shrinking the space to 0 and 1 without changing the original characteristics of the sample data. Normalization method is shown as follows.

$$S_{ij} = \frac{d_{ij} - d_j^{\min}}{d_j^{\max} - d_j^{\min}} \qquad (1)$$

In equation 1,i = 1,2, ..., m and j = 1,2, ..., n. m is the number of rows of the matrix. n is the number of columns of the matrix. djmin is the minimum value of the j-th column elements in matrix A. djmax is the maximum value of the j-th column elements in matrix.

2.3.2Creation of the fuzzy similar matrix

The fuzzy similar matrix is used to store similarity of different data, whose value ranges from 0 to 1. In this thesis, it is used to calculate the similarity of software defect data. Thus, when the normalized matrix is acquired, fuzzy similar matrix R can be set up using the maximum and the minimum method.

$$r_{ij} = \sum_{k=1}^{n} (s_{ik} \wedge s_{jk}) \Big/ \sum_{k=1}^{n} (s_{ik} \vee s_{jk}) \qquad (2)$$

2.3.3Creation of fuzzy equivalent matrix and extraction of relevant defects

In this thesis, fuzzy equivalent matrix T is established using transitive closure method. The close relationship between two variables is acquired by constant squaring method according to the principle of shortest path. When it satisfies the equation of T2k=Tk=T, a fuzzy equivalent matrix T is established. The similarity between any defect and other defects can be acquired from this matrix.

The low similarity between two defect data indicates that the difference between the two defects is large. Then it can be determined that there is no relationship between these two defects. Thus there is no need to proceed with the subsequent analysis. Therefore, according to the minimum similarity threshold set by the user, the software defect data whose similarity is greater than the threshold value and less than 1 can be extracted as a basis for the subsequent data analysis.

2.4Data matrix generation and Simplification.

Given item sets I composed of d items, I={i_1,i_2,…,i_d}. Given a set T composed of N transactions, T={T_1,T_2,…,T_d }. Its corresponding data matrix construction method is as follows.

$$a_{ij} = \begin{cases} p_{ij} & i \neq m \wedge j \neq n \\ num(p_{ij} \neq 0) & i = m \vee j = n \\ 0 & i = m \wedge j = n \end{cases} \qquad (3)$$

In equation 3, m=N+1, n=d+1. pij is the discrete coded values of items. Each row represents a transaction and each column represents an item. Function num is used to calculate the number of non-zero elements of each row or each column in the matrix. Therefore, the last line of the matrix is the number of each item appearing in the database and the last column shows the number of items in each transaction.

In order to improve efficiency of the analysis, the data matrix can be simplified by deleting columns containing infrequent item sets and rows containing only one item. Its process is as follows:

According to the given support threshold named minsup, the minimum number of frequent item sets appearance can be calculated. The calculation equation is as follows:

$$n = ceiling(\min\sup \times N) \qquad (4)$$

N is the total number of transactions, function ceiling is used to calculate the smallest positive integer not less than variable x.

Then comparison between n and elements amj in the last row of matrix A is done. If amj<n, it indicates that the number of items occurrence in the j-th column is less than the least occurrence number of frequent item sets. So it is a non-frequent item sets and column j of the matrix can be deleted. The row containing only one item can be derived from the value of the last column. After the simplification, the last row and column can be deleted.

2.5Extraction of negative association rules

In this thesis, the genetic algorithm is introduced to mine negative association rules. Firstly an initial set of rules are generated. Then the initial rules are optimized using a variety of evolutionary criterion to get the rules meeting the optimization criteria. Since the data matrix established above can be valued as an abstract representation of the original association characteristics of the defect data, so

the initial population of genetic algorithm can be acquired directly from the data matrix. After a multi-generation search of genetic algorithm, all the strong association rules can be acquired. But as for the negative association, due to its different form including X→Y, X→Y, X→Y, it can't be decided which form it is simply by chromosomes. Thus, the last work of this approach is to evaluate the association rules and extract the specific form of negative association rules[5].

From the definition of negative association rules, we can know that when a confidence threshold named minconf and a support threshold named minsup are given, if it satisfies the inequality constraint like c(¬X→¬Y)≥minconf and s(¬X∪ ¬Y)≥minsup or c(X→¬Y)≥minconf or c(¬X→Y)≥minconf, the form of the corresponding negative association rules can be determined.

For example, to chromosomes 10050, given minsup=0.25, minconf=0.5, confidence value can be calculated according to equation follows:

$$c(X \to \neg Y) = 1 - c(X \to Y) = 1 - s(X \cup Y)/s(X) \qquad (5)$$

$$c(\neg X \to Y) = \frac{s(Y) - s(X \cup Y)}{1 - s(X)} \qquad (6)$$

$$c(\neg X \to \neg Y) = 1 - c(\neg X \to Y) \qquad (7)$$

c(X→Y) is the confidence. s(X)is the support and can be calculated according to equation follows:

$$s(X) = \sigma(X)/N \qquad (8)$$

σ(X) is the count of support，which express the time number of item X appears in the project database. N is the total number of projects.

Results can be calculated as:

i) c(15→¬6)=1-(0/5)/(3/5)=1>minconf=0.5，so negative association rules15→¬6 can be extracted.

ii) c(¬15→¬6)=1- c(¬15→6)=1-1=0<minconf=0.5，so negative association rules¬15→¬6 can not be extracted.

3.Software testing projects optimization

Software testing work can be optimized according to the extraction results acquired through data analysis of software defects using the above method. Test schemes and test cases less prone to defects occurrence can be deleted. Besides, test cases that prone to defects occurrence should be strengthened. In this way, testing efficiency and accuracy can be improved. Based on the type of negative association rules, specific adjustment method are divided into the following three conditions:

i) For X→¬Y form of negative association rules, when test items containing defect X are completed and defect X happens, then test cases containing defect Y can be cut.

ii) For ¬X→¬Y form of negative association rules, when test items containing defect X are completed and defect X doesn't happen, then test cases containing defect Y can be cu.

iii) For ¬X→Y form of negative association rules, when test items containing defect X are completed and defect X doesn't happen, then test cases containing defect Y should be enhanced.

4.Application Examples

In order to illustrate GMNAR method clearly, taking software defect data generated in XX information management software testing process as the experimental data, the vector space model is constructed and the similarity is calculated. And one group of related items is shown in Table 2.

Table 2:Software defect and test items

Id	Related test items
AZH-V1.0-WBG-001	test item 1, test item 4, test item 5
AZH-V1.0-WBG-002	test item 2, test item 3, test item 6
AZH-V1.0-WBG-003	test item 1, test item 2, test item 3, test item 6
AZH-V1.0-WBG-004	test item 2, test item 5, test item 6
AZH-V1.0-WBG-005	test item 1, test item 2, test item 3, test item 5

Properties of this test item are encoded. Vp={test item 1, test item 2, test item 3, test item 4, test item 5, test item 6}. They are encoded as {1,2,3,4,5,6}. First components of negative association rules are named as X, X={test item 1, test item 2, test item 3, test item 4, test item 5}. Second components of negative association rules are named as Y, Y={test item 6}.

Its discretization data is shown in Table 3:

Table 3:discretization data

No.	X					Y
	1	2	3	4	5	6
1	1	0	0	4	5	0
2	0	2	3	0	0	6
3	1	2	3	0	0	6
4	0	2	0	0	5	6
5	1	2	3	0	5	0

Its data matrix *A* is as follows:

$$A=\begin{pmatrix} 1 & 0 & 0 & 4 & 5 & 0 & 3 \\ 0 & 2 & 3 & 0 & 0 & 6 & 3 \\ 1 & 2 & 3 & 0 & 0 & 6 & 4 \\ 0 & 2 & 0 & 0 & 5 & 6 & 3 \\ 1 & 2 & 3 & 0 & 5 & 0 & 4 \\ 3 & 4 & 3 & 1 & 3 & 3 & 0 \end{pmatrix}$$

Given *minsup*=0.25, *minconf*=0.75, then simplified matrix *B* is as follows:

$$B=\begin{pmatrix} 1 & 0 & 0 & 5 & 0 \\ 0 & 2 & 3 & 0 & 6 \\ 1 & 2 & 3 & 0 & 6 \\ 0 & 2 & 0 & 5 & 6 \\ 1 & 2 & 3 & 5 & 0 \end{pmatrix}$$

Original chromosomes C can be acquired as:

C={10050,02306,12306,02056,12350}.

Finally, through several Genetic operation, rules aggregate as:

C'={00050,02306,02006,00300,10050}.

At last, negative association rules is extracted as:

NegativeAR={15→¬6, ¬2→¬6}.

According to these negative association rules, software testing project can be optimized as follow:

i) According to rule 15→¬6: if one class defect is detected both in testing item 1 and testing item 5, there will be a high probability of nonoccurrence of this defect in testing item 6. Then testing design for item 6 can be cutout in follow testing.

ii) According to rule ¬2→¬6: if one class defect is not detected in testing item 2, there will also be a high probability of nonoccurrence of this defect in testing item 6. Then testing design for item 6 can

be cutout in follow testing.

5.Experiment results analysis

In this thesis, the author uses Matlab2012a to simulate the Apriori method and GMNAR method proposed in this paper.

To compare the credibility of the results, the simulation data are generated by using IBM's experimental data generator. In this experiment, four basic experimental databases are as follows: T16.I5.D0.5k, T130.I12.D6k, T1400.I130.D56k, T15000.I1400.D570k. Result is shown in Fig.1.

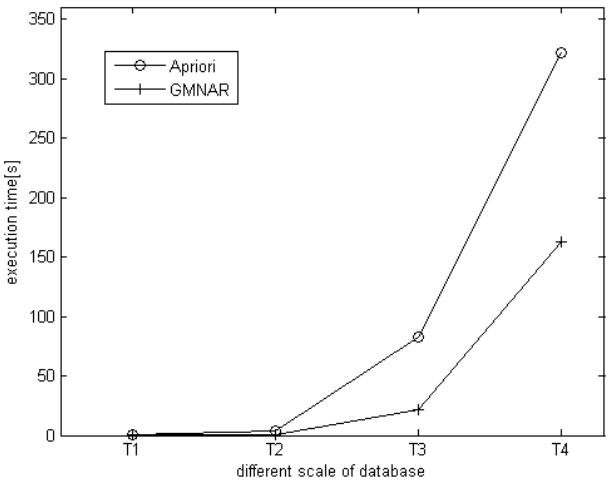

Fig.1 Execution time comparison of two methods operated in different scale of database

Fig.1 shows that for small database, the execution time of two methods is nearly same, but with the increase of size of the database, the execution time of GMNAR is significantly less than that of Apriori. GMANR only scans the database once to get the data matrix, while Apriori requires multiple scanning the database. When the database scale is relatively small, the running time gap between two methods is not obvious. With the increase of data size, GMNAR method has obvious advantage.

6.Conclusion

In this paper, a software testing optimization method based on negative association rules named GMNAR is proposed. And taking XX information management software testing defect data for example, it is described in detail. The method introduces the idea of vector space model and the data matrix to achieve negative association rules extraction and test items optimization. To some extent, it can improve the efficiency of software testing.

References

[1] Z. Honglei, X. Zhigang, L Ming. Study on negative association rules mining algorithms. Microelectronics and Computer, vol. 27(2010), p.167-169.

[2] D. Xiangjun, W. Shujing, S. Hantao, et al. Study on negative association rules mining methods. Beijing Institute of Technology, vol. 24(2004), p. 978-981.

[3] M. Zhanxin, L. Yuchang. Frequent item sets explosion problem in negative association rules mining. Tsinghua University (Natural Science), vol. 47(2007), p. 401212-1215.

[4] W. Guoyin. Rough set theory and knowledge acquisition. Xi'an Jiaotong University Press(2001).

[5] J. Cohen. Statistical power analysis for the behavioral sciences (2nd ed). Lawrence Erlbaum, New Jersey(1988).

[6] H. Xiong, et al. Exploiting a support-based upper bound of pearson’s correlation coefficient for efficiently identifying strongly correlated pairs. The 10th Intl.Conf. on Knowledge Discovery and Data Mining.Seattle,WA,2004 8:334-343.

[7] Brin S, et al. Beyond market baskets: Generalizing association rules to correlations[C]. Proceeding of the 1997 ACM SIGMOD International Conference on Management of Data. ACM Press, 1997:265-276.
[8] Agrawal R, et al. Mining association rules between sets of items in large databases[C]. Proceeding of the 1993 ACM SIGMOD International Conference on Management of Data. ACM Press, 1993:207-216.
[9] L. Tangping. Research and improvement of the association rules algorithm and the Apriori algorithm based on matrix. Xinan Jiaotong University, master degree paper, 2011
[10] Z. Yuquan, C. Geng, Y. Hebiao. Research of positive and negative association rules mining algorithm. Computer Science, 2006, 3 (3):188-190.