Research on Generation Methods of Test Cases for Pair-wise Covering Combinatorial Test Cases

He Dandan¹, Zhai Yue¹, Wang Lijuan¹
¹Dalian University of Science and Technology, Dalian 116026, China
²hedandan@dlust.edu.cn

Abstract: For most software systems, pair-wise combinatorial testing is highly practical and effective. This article mainly starts with the specific research status of the pair-wise covering combinatorial test and the main content of the software test, analyzes and compares the existing main implementation algorithms, and then evaluates their specific performance.

1. INTRODUCTION
As an important link in the software development process, software testing is an important means to ensure software quality and improve reliability. Combinatorial testing has been extensively studied at home and abroad, and it can adopt a test software system with fewer test cases under the premise of ensuring the error detection rate. The pair-wise combinatorial covering test not only considers the various factors currently existing in the system, but also considers the various factors that may have an impact due to the interaction. Therefore, the pair-wise combinatorial covering test can use the least test case data, and solve the related problems in the test process according to the specific needs of the software.

2. MAIN RESEARCH METHODS OF COVERAGE TESTING

2.1. Boundary Value Analysis
Boundary value analysis is used to test the data on the boundary of the ordered equivalence class. There are two boundary value analysis methods, including binary test method or three-value test method. The binary test method takes a boundary value (the value just on the boundary) and a value that just exceeds the boundary (the smallest possible increase). The coverage of boundary value analysis is defined as the total number of boundary conditions tested and the total number of boundary conditions identified. Boundary value analysis can be used for numerical range, numerical characteristics of non-numerical variables (such as length), data structure stored in loops (including loops in use cases), the accuracy of the analysis of active boundary values determined by time depends on the precise division of price categories. The boundary value analysis method must pay attention to the increase of valid data and invalid data in order to accurately define the test data.

2.2. Cause and Effect Diagram (CEG)
The cause and effect diagram can be derived from various resources describing the functional logic (rules) of the program. Such as from user stories (requirements are presented in the form of user stories), flowcharts, etc. The cause and effect diagram helps to obtain a block diagram of the logical structure of the program, and is typically used as the basis of a decision table. Using causality diagrams and
decision tables to capture allows the software to systematically achieve the test coverage required by the program logic.

Coverage = use case coverage of cause-to-effect lines / all possible cause-to-effect lines including a combination of conditions. CEG applies the same test level and case as the decision table. In particular, CEG shows the result caused by the combination of conditions and the result of exclusion. However, the use of CEG in the test requires more time and effort, the support of tools, and the designer and the user must understand the special symbols in the diagram [1].

2.3. Combinatorial Test Method
Within the scope of various testing methods for software testing, combinatorial testing is a method that generates test cases based on combined coverage criteria. The main idea of this method is to use as few test cases as possible to cover as many system factors as possible. When the software is tested, these influencing factors can be assumed to be multiple parameters, and each parameter has multiple values. The test results found that it is difficult to find errors when using the combined test method to test the system, because most of the software failures are caused by the interaction of various influencing factors.

For the purpose of illustrating the method of combinatorial testing, this paper selects the ‘calculator’ attachment of the PC operating system as an example.

Table 1. Examples of composite test cases

| Test parameter | Test Case 1 | Test Case 2 |
|----------------|-------------|-------------|
| ①Integer testing | 1+1        | ...         |
| ②Decimal test | 1.0+1.0    | ...         |
| ③Combination of keyboard tests | ... | ... |
| ④Combination of four operations | ... | ... |

Obviously, the ‘calculator’ test cases appear in an explosive state, and these endless test cases do not need to be executed in the actual test process. The most feasible way is to select typical test data from these test cases. In order to improve the test efficiency, the test cases can also be combined. This combination happens to cover a certain system test scenario. For example, to test the time and date (year, month, day, hour, minute, second, etc.), it is not necessary to test the six parameters separately, and it can be turned into a combined test problem with six variable parameters. Each variable will have a set of typical test values, and any combination will constitute a possible test case.

3. THE MAIN RESEARCH ALGORITHM OF PAIR-WISE COVERING COMBINATORIAL TEST

3.1. Orthogonal Latin Square Algorithm
Orthogonal Latin square algorithm, that is, an actual algorithm that fully utilizes the special properties of the Latin square. According to the algorithm, pair-wise combination is implemented to construct a coverage test set.

For example, in each row and column, each element of the Latin square can only appear once in the entire combination, so it can be regarded as a square matrix with special properties. The Latin side can effectively construct and cover the test case set of two combinations of all parameters based on the above-mentioned special properties, and form a single test case on this basis. On the surface, this method is very convenient and feasible, but the overall efficiency is relatively low. If the system has more than three parameters, it cannot be fully applied. The orthogonal Latin square algorithm can make full use of its special properties and orthogonal table, a small number of experiments, and finally a more clear and reliable conclusion. It only needs to follow a certain method to calculate the orthogonal table,
and the corresponding test cases can be generated during the generation process of the combined coverage test case of the entire team. Next, the operator must understand that the orthogonal Latin square algorithm mainly depends on the orthogonal table[2]. Even as a more effective use case selection method, there are still many problems to solve. For example, there is currently no better construction method for hybrid orthogonal tables. Therefore, in the selection of black box test cases, this algorithm has relatively large application limitations.

3.2. Williams Algorithm

The Williams algorithm was proposed by the University of Ottawa in 1997. The algorithm is mainly used in software application systems to test pairwise combination coverage strategies. Relevant practical research has proved that the application of this algorithm can save a lot of test resources. After entering the new century, this combined coverage idea has been widely used in the configuration and combination of multiple test software systems with an important role. Therefore, with the rise of component-oriented development technology at the beginning of this century, related researchers began to explore the interactive test coverage between components.

For example, related researchers believe through research that orthogonal Latin squares can be used as the basis of Williams’ algorithm. For three n-th Latin squares, the specific conditions of the three parameters of the n level can be properly processed, as shown in figure 1. Then take out the row number and column number of a single value as the specific corresponding position value, and then perform superposition and two-tuple matrix calculation by orthogonal Latin square. Assuming that the test case data is (3,2,1), the 3×3 orthogonal Latin square is shown in figure 1:

|   | 1 | 2 | 3 |
|---|---|---|---|
| I |   |   |   |
| II|   |   |   |
|   | 1 | 2 | 3 |
|   | 2 | 3 | 1 |
|   | 3 | 1 | 2 |

Figure 1. Latin grid system of three horizontal parameters

Then, the operator also needs to construct a set of orthogonal Latin squares applied to a system with more than three parameters, and then superimpose them. For example, if there is a system with m parameters, then calculate the number of orthogonal Latin squares to be (m-2), and then combine the values of the corresponding positions in the (m-2) orthogonal Latin squares to make a (m-2) tuple, so that m parameter values can appear. These parameter values appear when the test data is generated, corresponding to the row number, column number, and (m-2) element of a single tuple group [3].

|   |   |
|---|---|
| I | II |
| 1 | 2 | 3 | 1 | 2 | 3 |
| 2 | 3 | 1 | 3 | 1 | 2 |
| 3 | 1 | 2 | 2 | 3 | 1 |

Figure 2. Two orthogonal Latin grid systems with three levels and four parameters

According to the core of the W algorithm, if you have a three-level four-parameter system, then (4-2) orthogonal Latin squares are needed, as shown in figure 2. Therefore, the two orthogonal Latin squares can be superimposed first, and then the corresponding two-tuple superposition matrix can be obtained. Then the row number, column number and tuple value of the corresponding test generated data after the tuple is boxed are also fixed. The superposition matrix is shown in figure 3.

|     | (1,1) | (2,2) | (3,3) |
|-----|-------|-------|-------|
| (2,3)| (3,1) | (1,2) |
| (3,2)| (1,3) | (2,1) |

Figure 3. W Binary group superposition matrix of the algorithm
It can be seen from the above that the application of the orthogonal Latin square method requires that all parameter levels are consistent and each parameter is independent of each other. If each parameter has a different value, and there is more mutual influence between parameters, then there will be no suitable method for orthogonal Latin squares. There are also related researchers who have made certain improvements to this algorithm, but the specific research is not in-depth.

3.3. Kobayashi Algorithm
Kobayashi algorithm is an algebraic method given by experts such as Kobayashi and Tsuchiya. It is also an algorithm that constructs the basic orthogonal table and its feature block as the basic component, and realizes the combination coverage in pairs. The orthogonal table is the basic component, and the new pairwise composite overlay table used is constructed by the cross overlap pattern of the orthogonal table.

For example, a certain multi-dimensional factor system of prime powers is an important basis for the algorithm to calculate the number of values. If the number of parameters (factors) is less than the sum of prime powers and 1, there must be an orthogonal table, so it is easy to generate such an orthogonal table. After the orthogonal table is generated, a large-scale promotion can be carried out on this basis. Then, when the number of parameters is greater than the sum of prime powers and 1, the orthogonal table and the block structure table can be used to construct. If the number of selected values is not a prime power, the relevant technical personnel must find a minimum prime power greater than the number of values to perform corresponding approximation processing. The same is true for the design of multiplicity, and it is also necessary to find a maximum prime power that cannot exceed the number of values or not less than the minimum prime power to generate an orthogonal table. Based on this, multiple paired coverage tables are generated. Next, the operator will combine the two situations flexibly in most cases.

Therefore, to a certain extent, the final calculation effect of the algorithm is the error when the selected prime power approximates, because this error is relatively large in many cases. For example, consider generating 64 needs to find the power of the smallest prime number greater than 6. It needs to calculate how many rows are needed to generate 74. This can solve the problem of large errors very well, and analyze the specific situation also makes the test results more objective, which is more helpful for the next test link.

3.4. AETG Algorithm
This algorithm was originally proposed by relevant researchers at Bell Labs in the United States, and is mainly a specific method based on the heuristics of pairwise combined coverage test data. This method can be used to generate test data according to actual test requirements, and then realize the pair-wise combination coverage of the entire system parameters, or realize the combined coverage test of multiple parameters.

For example, when applying the algorithm, a set of candidate use cases must be generated first, and these use cases need to be calculated according to the greedy algorithm. On this basis, choose the one that can cover the most uncovered paired combination use cases. This algorithm has been used for a long time in some areas, and it can also generate corresponding test plans. For example, it can effectively monitor the consistency test of the communication protocol, and can also test the network monitoring system. Some Chinese scholars have also fully applied the algorithm tool in the testing of the PLA command training information system through research. When conducting a single-round test command training information system, the application coverage criteria can detect an error detection rate of about 70%. Using this algorithm for repeated testing can effectively improve the detection rate of error detection.

3.5. IPO algorithm
IPO algorithm is proposed by the researchers of the computer department of the University of North Carolina. This algorithm needs to strictly follow the progressive extension method of parameter order.
On this basis, a pairwise combination coverage test data generation method is obtained. For example, first of all, this algorithm needs to take parameters as the main research object. At the beginning, it will generate a set of use cases that meet the requirements of pairwise composite coverage test, and then extend the remaining parameters one by one. Until all parameters are included in the entire test case. Moreover, in the entire algorithm, the patient has to keep the size of the entire test set to the best possible state. Then, the parameters are expanded according to two specific steps, including horizontal expansion and vertical expansion. It needs to expand the optimal test case, and add the value of the parameter to be extended to the existing test case. Then, after the horizontal expansion, the remaining uncovered pairwise combinations are vertically expanded to form a new organization, thereby obtaining new test cases.

3.6. PSST Algorithm

PSST algorithm is a test case generation method based on the combined coverage test model. For example, the data used for testing is treated as a solution space tree, and then the test data is supplemented by greedy algorithm. By using this method, all the solutions with no more than 1 overlap need to be found out, and then extended accordingly.

![Figure 4. Examples of logical overlay for solving space trees](image)

4. ANALYSIS AND COMPARISON OF ALGORITHMS

The above-mentioned algorithms adopt different methods when constructing the corresponding test data. Although the basic composition of the above algorithm is slightly different, it still has a certain versatility. It is reflected in the analysis of specific problems. If different methods can be effectively combined, then better practical results will be produced. But these test methods are still in the category of traditional test methods in the final analysis. Therefore, it needs to first analyze their shortcomings to improve test efficiency and obtain more accurate results.

For example, the advantage of the first orthogonal Latin square algorithm is that the structure is very
simple and there are mathematical models in reality. If you want to accurately calculate the size of the test set that can meet the coverage requirements of the established combination, you only need to know the number of known parameters and the specific values. In order to fully satisfy the mathematical characteristics of the Latin square, no official positions need to be added to the actual construction algorithm, so that additional test cases can be effectively increased. The second algorithm takes the orthogonal Latin square algorithm as an important basis, adding factors such as partitions into the orthogonal matrix. In the third algorithm, the main basic components include the basic orthogonal matrix and the feature block, and on this basis can the requirements that conform to the paired comprehensive coverage table be constructed. The fourth algorithm can get the best results, but it can't compare to the IPO algorithm in terms of time complexity. The fifth algorithm is mainly based on the method of establishing a combination of test cases that are gradually expanded in the order of parameters. This algorithm takes parameters as the center, which will have a certain impact on the efficiency of the IPO algorithm. The sixth algorithm mainly explains the space mapping into a tree, and then finds out all the solutions that do not overlap with each other more than one, and then expands them accordingly. This algorithm mainly combines the comprehensive advantages of algebraic and heuristic methods, and effectively overcomes the shortcomings. It not only allows testers to test a specified set of test cases, but also expands accordingly to generate a paired combination coverage table. The algorithm is still a heuristic algorithm, and there is no guarantee that the test results are optimal every time.

REFERENCES:
[1] Hui Zhanwei, Huang Song, Su Shihan, etc. Research on the generation method of pairwise covering combination test cases [C]. 2010 Asia-Pacific Conference on Information Theory, 2010: 264-267.
[2] Li Longshu, Wang Yating. Pair-wise combination test case set generation algorithm with weight parameters [J]. Computer Engineering, 2015(4): 284-287.
[3] Yan Jun, Zhang Jian. The principle and method of combinatorial test [J]. Journal of Software, 2009(20): 1393-1405.
[4] Tai K CLei Y, A test generation strategy for pairwise testing IEEE Transactions on software Engineering, 2002, 28(1): 10911