Research on Multi-constraint Combinatorial Test Technology for High Confidence Embedded Software

Feng Gao ,Fei Deng,Yunqiang Yan
Institute of Computer Application
China Academy of Engineering Physics
MianYang,China
achang85@163.com

***Abstract*—The importance and complexity of software in embedded devices are increasing. It becomes an active research topic on how to carry out the full and efficient testing of military high confidence embedded software such as weaponry, aerospace and so on. The combinatorial test is an effective test case generation technique. But the complexity of the constructor of the combinatorial test suites is NP-complete. The effectiveness and complexity of combinatorial test methods have attracted researchers in the field of combined mathematics and software engineering to study it deeply. For the characteristics of military embedded software, a multi-constraint combinatorial test method is proposed. This method takes into consideration the constraint relationship among parameters, and uses it to guarantee the covering of the seed combination and the simplification of the use case set. And it implements the test case generation tool based on this method. The results show that this tool generates a few use cases and can guarantee the covering of key combinations.**

Key Words: High Confidence; Embedded Software; Multi-constraint; Combinatorial Test

I. INTRODUCTION

With the development of information technology, the importance and complexity of embedded system are higher and higher. The software caused by the embedded system fails covers about 70%[1-2];Software becomes an important factor restricting the quality of embedded system. Software testing is an important means to ensure software quality. It becomes an active research topic on how to carry out the verification and verification (V&V) of high - efficient and high - quality weapon equipment, aerospace and other high confidence embedded software[3-6]. A lot of practice shows that combinatorial testing is an effective software testing method. It generates a small amount of high-quality test data and systematically tests the combination of parameters. Testing case set generation is a hot topic in combinatorial testing research. People have presented many mathematical methods, heuristic search methods and various greedy algorithms. But these methods have their own limitations and can only have certain advantages when solving some certain problems. For example, TConfig[6] is a mathematical method to construct the recursive structure by using the orthogonal tables and other basic components. This method has a fast generating speed. But the downside is that it need to rely on existing algebra or combined objects. For the heuristic search, we can use tabu search [7], simulated annealing (SA)[8] and other methods to generate small test case sets. But these methods generally take a long time. In contrast, heuristic greedy algorithms are not only flexible but also fast. At present, a variety of heuristic greedy algorithms have been presented such as AETG[9-11], etc.TCG[12], DDA[13-14] and IPO[15]. Each of these methods has its own advantages. They are all universal methods which are not ideal for solving some specific problems.

DOI reference number: 10.18293/SEKE2019-016

In this paper, according to the characteristics of military high confidence embedded software such as weapon equipment, aerospace and so on, a multi-constraint combination test method which can configure seed combinations, covering strength and parameter constraints is proposed. This method takes into consideration the importance of different parameters and the constraint relation between the parameters, taking advantage of them to ensure the covering of the seed combinations and the simplification of the use case set. And it implements the test case generation tool based on this method. The test result shows that this tool generates a few use cases and can guarantee the covering of key combinations.

The second section introduces the basic concepts and models of combinatorial test. The third section presents the multi-constraint combinatorial test methods. And the fourth section describes the experimental design and the results, finally the summary and outlooks.

II. THE BASIC MODEL OF COMBINATORIAL TEST

Assume n parameters are affecting the system software SUT(software under testing), recorded as $F = \{f_1, f_2, \cdots, f_n\}$. These parameters can be SUT configure parameter internal events external input, etc. The parameter f_i has a_i possible values by equivalence partitioning, boundary value method and other pre-parametric decomposition, forming the value set $V_i = \{a_1, a_2, \cdots, a_i\}, 1 \leqslant i \leqslant n$.

Definition 1. Regard n-tuple $(v_1, v_2, \cdots v_n)$ as a test case for SUT, and $v_1 \in V_1$， $v_2 \in V_2$， $\cdots$, $v_n \in V_n$.

Correspondingly, regard a collection of n tuples as a test case set for SUT. In the combinatorial test, the test case set is often referred to as a combination covering table (Covering Array for short).

Definition 2. The t-dimension overlay table of the system SUT $CA(N;t,n,(v_1, v_2, \cdots v_n,))$ is an N*n array. The i column corresponds to the i parameter, which is recorded as v_i. The subarray of N*t that is formed by any t parameter contains all

of the t tuples of the t parameter, in which t is the strength of the combinatorial test.

According to the definition of the covering array CA(covering array) which is given by Cohen et al. to describe the set of test cases in a combinatorial test case, generally, an array of overrides that can cover any two parameters of any parameter is called a two-dimensional combination (2-way) or Pairwise combination. t-way is the combination of values that can cover the t parameters. When t is equal to the number of parameters n, the override array can override all of the combined conditions of the parameter system, which is 100% full coverage. Therefore, the mathematical model of the combinatorial test case set is often non-continuous, multi-objective and nonlinear constraint solving the problem.

III. MULTI-CONSTRAINT HCESCT COMBINATORIAL TEST METHOD

The most important difference between military embedded software and general software is that the security and reliability requirements of military embedded software are extremely high, and it is best to have 100% test case coverage. But some special system software that has a combination explosion or an important level of software can't reach 100% complete coverage because of many considerations for military software testing, the complexity of operating environment, time and cost. Reasonable use of combinatorial test technology for the military embedded system software to select effective test case set can make up the randomness of artificial design cases and randomness of test cases.

HCESCT（High Confidence Embedded Software Test Generation Method for Combinatorial Testing） is the combinatorial test case generation method presented by this paper with military embedded software as the test target. In combination with the characteristics of high reliability and security requirements, HCESCT method is mainly concerned with adopting reasonable strategies to solve the problem of explosion of cases in the test process. At the same time, ensure that important values are not missed. The combinatorial test method process in this paper is shown in Fig. 1.

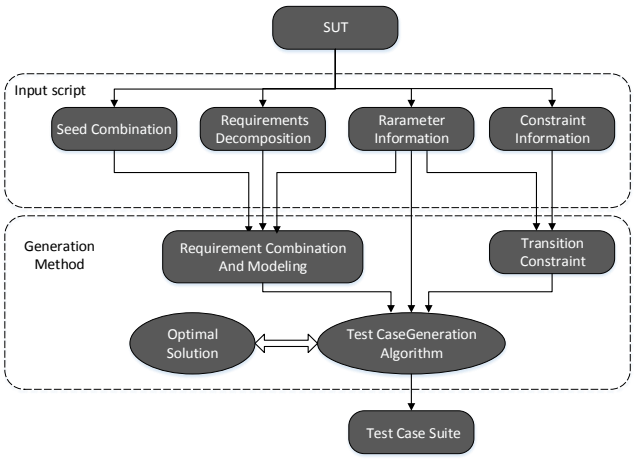

Fig. 1. The Combinatorial Test Method Flow Chart

According to the characteristics of high confidence embedded software, some key parameters must be covered. Depending on the importance of parameters, some require only 2-dimensional coverage, but some combination of parameters need to be covered in higher dimensions. In real software, there is always a certain dependence relation between the parameters of software, which leads to some constraint relationship among some values in these parameters. In order to support the various flexible strategies of the military embedded software combinatorial test method, important combinations must be guaranteed not missing. Here are a few effective generation strategies.

A. Seed Combinations

In high confidence embedded software with high-reliability requirements, it is particularly important to have some parameter combinations. If these parameter combinations go wrong, they can have disastrous consequences. Therefore, the Seed combination (Seed) strategy is added. At the stage of software portfolio testing modeling, add the combination of parameters that must be combined in the software application scenario as a seed combination to ensure that the use case set generated at the end of the HCESCT algorithm is sure to screen out these important combinations. The modeling approach is as follows:

[SEED]

P1:v1,P2:v2,P3:v3;

This constraint statement (P1,P2,P3)=(v1,v2,v3) must be present in the test case

B. Variable Intensity Coverage

Also in order to ensure the coverage of the tested software, the coverage of about 70% of the matched pairs is obviously poor. In multi-parameter systems such as influencing factors, software input and pattern categories, it is necessary to adopt a flexible and variable strategy to ensure the high-dimensional combination of important parameters and the secondary parameter matching combination. For example, on a loaded software, functional requirements specify that the most important parameters affecting software output include the launch mode parameters Mod, temperature Tem, wind power WindF and the wind condition WindD. In parametric system modeling, the variable intensity coverage strategy should be used to cover the four parameters in a 3-way or 4-way combination. At the same time, other parameters with little effect on the launch function such as humidity, launch time and altitude use a matching combination. Enter in the modeling script:

[STRENGTHS]

default : 2; // The global coverage intensity is matched.

Mod，Tim，WindF，WindD : 3； // Variable intensity, important parameter high dimensional combination

With some scripts as input, this algorithm combines the parameters of different important levels with varying intensity.

C. Parameter Constraint

In military software， there’re always situations that signal A is received, but parameter b-e fails or signal B is received, but parameter M-N fails. This situation is called a

conflict among parameters. To solve this problem, the parameter constraint description is added to the script during the parameter modeling phase. Adding the parameter constraint strategy, while searching in the generation algorithm, for any parameter:

- Collect all the conditions that block this parameter;
- The parameter is blocked when any condition is satisfied;
- This parameter must be valid when all conditions are not satisfied.

In accordance with the above judgment logic, the algorithm is able to automatically block the combination of invalid parameters that have been identified, and prevent the algorithm from generating too many invalid test cases. The parametric constraint modeling method is as follows:

[CONSTRAINTS]

P1==v1->P2!=v2;

P3==v3->P4==v4& P5>v5。

D. HCESCT Algorithm Introduction

As mentioned above, HCESCT uses the greedy algorithm that incorporates a flexible and practical combined strategy. Adopt the classic line-by-line search and a one-time-one-line generation strategy:

1) Initialize covering requirements（Target combination: a combination that needs to be covered）

- Intensity—combination of all two parameters

2) Generate test cases per article

- Try to cover many of the uncovered target combinations.
- The new test case must satisfy all constraints.
- Combinatorial optimization problem

3) When a new target combination cannot be overridden, stop.

- The remaining uncovered target combination violates the constraint and cannot be overridden.

The methods of constraint in the algorithm are as follows：

- Convert to a forbidden combination
- The constraint solving technique is adopted to ensure the correctness of the results.

The algorithm framework is shown below：

TABLE I. ALGORITHM KERNEL ALGORITHM OF HCESCT

Algorithm Kernel algorithm of HCESCT

```
1: init(combination_set);
2: while true do
3:   gen_opt_problem();
4:   if solve() == OK then
5:     new_test_case= translate_solver _output();
6:     test_suite.add(new_test_case);
7:     update(combination_set, new_test_case);
8:   else
9:     break;
10:  end if
11: end while
```

The traditional greedy algorithm produces a new test case that determines and overrides the uncovered combination in the while. This is an optimization problem for both search methods and heuristic methods. There are some algorithms for handling parameter constraints. Distribution parameter values in the algorithm, for example, they call external solver or other methods to determine whether the test cases to satisfy the constraints, or to determine which parameter values can be assigned to the parameters in recycling. In contrast, our algorithm integrates generation optimization and constraint solving together. In each while, every time a new test case is generated, only the optimization problem is considered to ensure that the new test case covers at least one uncovered combination. The judgment and retry process that generates the use case satisfies the constraints are done in the external solution function, not in the upper algorithm while. Because the constraint solving function solve the parameter constraint judgment problem very well, this will save a lot of resources in the overall cost of the algorithm. HCESCT algorithm diagram is shown in Fig. 2.

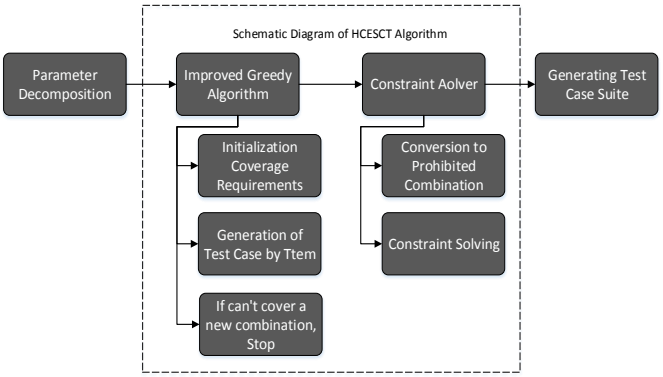

Fig. 2. HCESCT Algorithm Diagram

IV. THE COMBINATORIAL TEST EXPERIMENT AND THE GENERATION EFFECT CONTRAST

To illustrate the method of combinatorial test strategy in this paper and contrast the generation effects of existing main tools, some high confidence embedded data processing software are taken as an example. The software is in the demand analysis phase. And the type of input is shown in the table below:

TABLE II. INPUT DATA TYPE

	A Group P Data	**A Group M Data**	**B Group M Data**	**B Group M Data**
Data Source 1	0	0	0	0
Data Source2	0	0	0	0
Data Source3	0	0	0	0

Data processing software has three different types of data sources, and each data source is independent of each other. The software can accept data from three sources at a time. The data format of each data source is one of the four types of A group P data, A group M data, B group P data and B group M data. Among them, the A group P data of data source 1 and the B group P data of data source 3 is an important combination of data sources and the most frequently processed data of software. Similarly, data source 1 and data source 2 will not send B group P data to the data

processing software when the B group P data of data source 3 is available.

As is shown in HCESCT combinatorial test method flow chart, after this data is decomposed into embedded software requirements, a requirement combination is required. That means taking A group P data of data source 1 as a parameter value. The parametric model after modeling is shown in table 3.

TABLE III. COMBINATORIAL TEST PARAMETER

	A Group P Data	A Group M Data	B Group M Data	B Group M Data	No Data
Parameter V1	1_A_P	1_A_M	1_B_P	1_B_M	1_NO
Parameter V2	2_A_P	2_A_M	2_B_P	2_B_M	2_NO
Parameter V3	3_A_P	3_A_M	3_B_P	3_B_M	3_NO

In the data processing model of this instance, there are three input parameters. V1, V2 and V3 represent the different values of three data sources. The combination of these conditions is all data processing of the software being tested. To guarantee 100% covering rate, a full combination of three parameters requires 5*5*5, a total of 125 inputs. Assuming that in each input case, the combination of the four parameters needs to be tested, and each parameter range has three parameter values, then the final generated use case is 125*3*3*3*3, in total 10125 test cases. Because every data preparation and data processing test for this tested model weapon data processing software takes a long time and the testing of these test cases one by one consumes a lot of resources, it is far beyond the cost of the test.

The HCESCT combinatorial test method was used to analyze the demand of the measured parts and integrate and model the requirements. The seed combinations are: V1 = 1_A_P , V2 = 2_NO , V3 = 3_B_P. When the parameter constraint is V3 = 3_B_M, V2 != 2_B_M; When the parameter constraint is V3 = 3_B_P, V2！=2_B_P , V3 != 3_B_P. Use the script as shown in the following diagram to generate input for the combinatorial test case generation algorithm.

```
TestModel-INPUT

[PARAMETERS]
string V1:"1_A_P","1_A_M","1_B_P","1_B_M","1_NO";
string V2:"2_A_P","2_A_M","2_B_P","2_B_M","2_NO";
string V3:"3_A_P","3_A_M","3_B_P","3_B_M","3_NO";

[STRENGTHS]
default:2;

[SEEDS]
V1:"1_A_P",V2:"2_NO", V3:"3_B_P";

[CONSTRAINTS]
V3=="3_B_M"->V2!="2_B_M";
V3=="3_B_M"->V1!="1_B_M";

V3=="3_B_P"->V2!="2_B_P";
V3=="3_B_P"->V1!="1_B_P";
```

Fig. 3. The Combinatorial Test Case Generation Algorithm Input

The results generated by the combinatorial test case generation program and the PICT[16] execution results are shown in Fig. 4. There are 29 use cases generated by this method, and PICT generates 31 test cases. Furthermore, the syntax of this article is more concise, which avoids tools like PICT using complex syntax such as judgment statements to express parameter constraints.

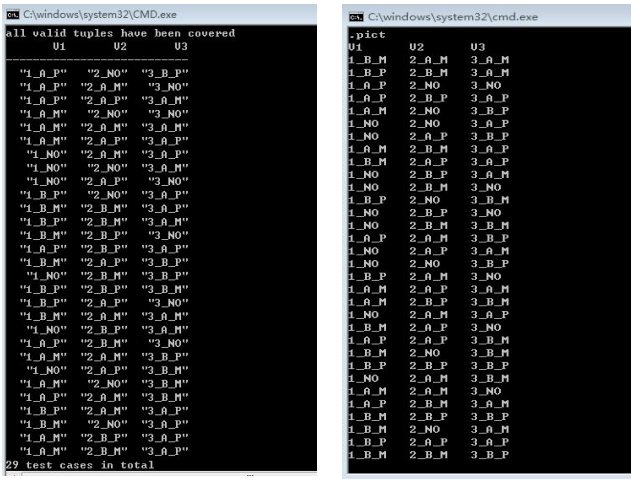

Fig. 4. The comparison of using effect between the method in this paper and PICT

V. SUMMARY AND FUTURE WORK OUTLOOK

In this paper, according to the characteristics of the military high Confidence embedded software testing, in the traditional high resource overhead test mode based on all matched pairs of parameter values, a line-by-line search method is used to Generate test case sets that cover a higher value and make up for the omission of tests that have been used to extract the quality of test cases. Secondly, this paper takes the use case generation strategy as the example of seed combinatorial parameter constraint variation and tries and successfully combines the combinatorial test algorithm, which provides more sophisticated means to improve the quality of test cases of embedded software in military. Finally, the use case generation tool based on this method is compared with PICT. The result shows that the method of this paper generates a few use cases and can guarantee the covering of key combinations. Further research on test case screening strategies can be strengthened from existing work. For example, in certain conditions, consider multiple parameters as one parameter to filter test cases in this granular manner.

REFERENCES

[1] Yang, Shunkun, and J. Fu. "A Model-Driven Testing Environment for Embedded Software." Energy Procedia 11(2013):1517-1525.

[2] Hopf, Michael, and J. Ovtcharova. "Integration of Virtualized Environments in PDM Systems for Embedded Software Product Development ☆." Procedia Cirp 11(2013):346-351.

[3] Wu, Yumei, Z. Yu, and Z. Liu. "Study of task profile oriented embedded software test aiming to improve reliability." International Conference on Future Computer and Communication IEEE, 2010: V2-58-V2-62.

[4] Yin, Yongfeng, Z. Li, and B. Liu. "Real-time Embedded Software Test Case Generation Based on Time-extended EFSM: A Case Study." Wase International Conference on Information Engineering IEEE, 2010:272-275.

[5] Wang, Yichen, X. Lan, and Y. Wang. "Modeling Embedded Software Test Requirement Based on MARTE." IEEE, International Conference on Software Security and Reliability-Companion IEEE, 2013:109-115.

[6] Williams, A. W., and R. L. Probert. "A practical strategy for testing pair-wise coverage of network interfaces." International Symposium on Software Reliability Engineering, 1996. Proceedings IEEE, 1996:246-254.

[7] Nurmela, Kari J. Upper bounds for covering arrays by tabu search. Elsevier Science Publishers B. V. 2004.

[8] Cohen, Myra B., et al. "Constructing Test Suites for Interaction Testing."International Conference on Software Engineering, 2003. Proceedings IEEE, 2003:38-48.

[9] Cohen, D. M, et al. "The AETG System: An Approach to Testing Based on Combinatorial Design." IEEE Transactions on Software Engineering 23.7(1997):437-444.

[10] Cohen, D. M, et al. "The Automatic Efficient Test Generator (AETG) system." IEEE (1994):303-309.

[11] Cohen, David M., et al. "The Combinatorial Design Approach to Automatic Test Generation." Software IEEE 13.5(1996):83-88.

[12] Tung, Yu Wen, and W. S. Aldiwan. "Automating test case generation for the new generation mission software system." Aerospace Conference Proceedings IEEE Xplore, 2000:431-437 vol.1.

[13] Colbourn, Charles J., M. B. Cohen, and R. Turban. "A deterministic density algorithm for pairwise interaction coverage." Iasted International Conference on Software Engineering DBLP, 2004:345-352.

[14] Bryce, Renée C, and C. J. Colbourn. "The density algorithm for pairwise interaction testing." Software Testing Verification & Reliability 17.3(2010):159-182.

[15] Tai, K. C., and Y. Lie. A Test Generation Strategy for Pairwise Testing. IEEE Press, 2002.

[16] Qin X, Duan J, Feng G, et al. Test Scenario Design for Intelligent Driving System Ensuring Coverage and Effectiveness[J]. International Journal of Automotive Technology, 2018, 19(4):751-758.