Algebraic strategy to generate pairwise test set for prime number parameters and variables

Conference Paper · September 2008

DOI: 10.1109/ITSIM.2008.4632002 · Source: IEEE Xplore

Mohammed I. Younis
University of Baghdad
48 PUBLICATIONS **351** CITATIONS
SEE PROFILE

Kamal Z Zamli
Universiti Malaysia Pahang
167 PUBLICATIONS **1,250** CITATIONS
SEE PROFILE

Nor Ashidi Mat Isa
Universiti Sains Malaysia
225 PUBLICATIONS **2,398** CITATIONS
SEE PROFILE

Algebraic Strategy to Generate Pairwise Test Set for Prime Number Parameters and Variables

Mohammed I. Younis
School of Electrical and Electronic Engineering
Universiti Sains Malaysia Engineering Campus
14300 Nibong Tebal, Penang, Malaysia
younismi@gmail.com

Kamal Z. Zamli
School of Electrical and Electronic Engineering
Universiti Sains Malaysia Engineering Campus
14300 Nibong Tebal, Penang, Malaysia
eekamal@eng.usm.my

Nor Ashidi Mat Isa
School of Electrical and Electronic Engineering
Universiti Sains Malaysia Engineering Campus
14300 Nibong Tebal, Penang, Malaysia
ashidi@eng.usm.my

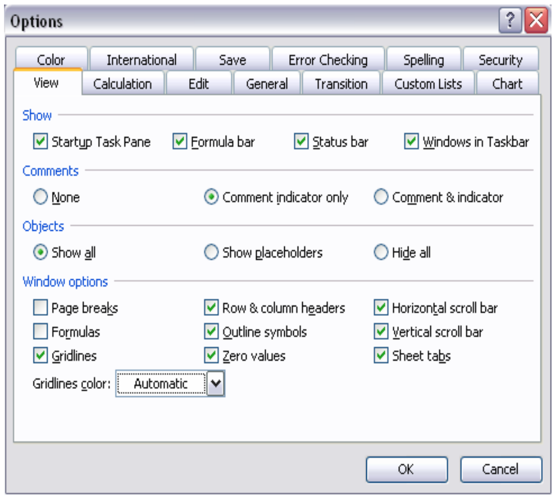
Abstract

Generating pairwise test set when the total number of variables is prime numbers has a remarkable property in that the test case generation process can be simplified by applying straightforward strategy that does not require any storage. This paper discusses the said algebraic strategy and compares the results with the well-known orthogonal array strategy. Additionally, this paper also demonstrates the applicability and simplicity of the strategy as compared to orthogonal array to obtain optimal test set for pairwise testing.

1. Introduction

Software testing relates to the process of executing a program or system with the intent of finding errors. Covering as much as 40 to 50 percent of the total software development costs, software testing can be considered one of the most important activities for software validation and verification. Lack of testing can lead to disastrous consequences including loss of data, fortunes and even lives. Many combinations of possible input parameters, hardware/software environments, and system conditions need to be tested and verified against for conformance based on the system's specification. Often, this results into combinatorial explosion problem.

As illustration, consider the option dialog in Microsoft Excel software (see Figure 1). Even if only View tab option is considered, there are already 20 possible configurations to be tested. With the exception of Gridlines color which takes 56 possible values, each configuration can take two values (i.e. checked or unchecked). Here, there are 2^{20}x56 (i.e. 58,720,256) combinations of test cases to be evaluated. Assuming that it takes 5 minute for one test case, then it would require nearly 559 years for a complete test of the View tab option.



Show
✓ Startup Task Pane ✓ Formula bar ✓ Status bar ✓ Windows in Taskbar

Comments
○ None ● Comment indicator only ○ Comment & indicator

Objects
● Show all ○ Show placeholders ○ Hide all

Window options
☐ Page breaks ✓ Row & column headers ✓ Horizontal scroll bar
☐ Formulas ✓ Outline symbols ✓ Vertical scroll bar
✓ Gridlines ✓ Zero values ✓ Sheet tabs
Gridlines color: Automatic

Figure 1. Option Dialog for Microsoft Excel

Similar situation can be observed when testing hardware product. As a simple example, consider a hardware product with 20 on/off switches. To test all possible combination would require 2^{20} = 1,048,576 test cases. If the time required for one test case is 5 minutes, then it would take nearly 10 years for a complete test.

As highlighted above, given limited time and resources, exhaustive testing is next to impossible [1].

978-1-4244-2328-6/08/$25.00 © 2008 IEEE

Obviously, there is a need for a systematic strategy in order to reduce the test data set into manageable ones.

Pairwise sampling strategy (i.e. based on two-way parameter interaction) has been known to effectively reduce the test data set (and yet be able to detect from 60 to 80 percent of the faults). To ensure acceptable coverage with optimum test data reduction, any two combinations of parameter values are to be covered by at most one test [2].

Many strategies do exist to address the aforementioned NP complete problem (e.g. Orthogonal Array [3], Automatic Efficient Test Generator (or AETG) [2], GA (Genetic Algorithm) and ACA (Ant Colony Algorithm) [4], In Parameter Order (IPO) strategy [5] [6], hill climbing and simulated annealing (SA) [7], and the Intersection of Residues Pair Set Strategy (IRPS) [8]). Building and complementing the aforementioned work, this paper discusses a remarkable property useful to simplify pairwise generation of test set. Here, a straightforward strategy without any storage requirement can be employed to facilitate the test generation process when the total number of variables and values are prime number. In doing so, this strategy is compared with the well-known orthogonal array strategy to demonstrate the optimal test set.

This paper is organized as follows. Section 2 gives an overview of orthogonal array. Section 3 describes the complete strategy used in this paper. Finally, section 4 gives our conclusions and some suggested future work.

2. On orthogonal array strategy

Orthogonal Array Strategy (OA) is a mathematical concept that has been known for quite some time [9]. The application of orthogonal arrays to testing was first introduced by Mandl [10] and later extended by Williams and Probert [11].

The main component of OA strategy is Latin squares. Here, a Latin square is a V x V matrix completely filled with symbols from a set that has cardinality V. The matrix has the property that the same symbol occurs exactly once in each row and column. As illustration, figure 2 depicts a 3 x 3 Latin Square consisting of a set = {0, 1, 2}.

0	1	2
1	2	0
2	0	1

Figure 2. A 3 x 3 Latin square

Two Latin squares are said to be orthogonal if each pair of elements occurs exactly once when each square is combined entry by entry. Figure 3 depicts the combination of two orthogonal 3x 3 Latin Squares.

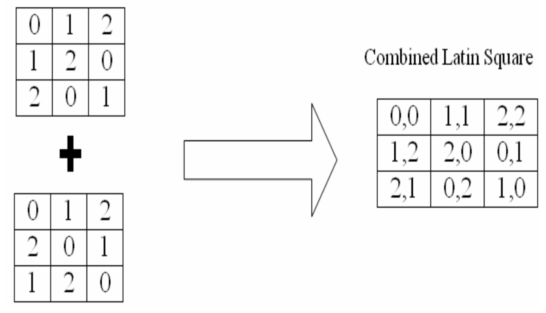


Figure 3. Two orthogonal 3x3 Latin square and the resulting combined square

If indexes are added to the r (ows) and the c (olumns) of the matrix, each position in the matrix can be described as a tuple < r; c; zi >, where zi represents the values of the < r; c > position. Figure 4 contains the indexed Latin Square from Figure 2.

Coordinates	c(olumns)		
r(ows)	0	1	2
0	0	1	2
1	1	2	0
2	2	0	1

Figure 4. Two ortogonal 3x3 Latin square augmented with coordinates

Figure 5 depicts the resulting set of tuples from the consideration of row, column and the values. Here, it should be noted that all tuples satisfy pair-wise coverage [12]. Interested reader can visit Sloan's website [13] for a predetermined OA of various size.

Tuples	Tuple < r,c,z_i >
1	000
2	011
3	022
4	101
5	112
6	120
7	202
8	210
9	221

978-1-4244-2328-6/08/$25.00 © 2008 IEEE

Figure 5. Tuples from two orthogonal 3x3 Latin squares that satisfy pairwise coverage

In order to understand the OA strategy using Latin square, it is necessary to generalize its essence. Here, for N (number of variables, each with N parameters), there is a need for to identify (N-2) distinct Latin squares. To identify N-2 distinct Latin square is straightforward for small values of N. However, if N is large (e.g. for N=23, we 21 distinct Latin squares), the generation of distinct Latin squares can be painstakingly difficult. In order to alleviate some of the abovementioned difficulty, we propose to a strategy that generates optimal number of test set when the number of parameters equal to number of variables, and both are primes numbers.

3. Proposed Strategy

In this section, we present a straightforward strategy that generates optimum test cases when N is a prime number. The strategy employs three nested loops. The n loop counts the number test case for each variables (i.e. equals to N), while i loop counts N test case for each n variables. Here, the j loop fills the parameter values, that is, for each j loop a test case is generated to cover N*N=N^2 test cases.

```
 for (int n=0;n<N;n++)
  begin
    for(int i=0;i<N;i++)
     begin
        for (int j=0;j<N;j++)
         begin
           fill test case with ((n+(i*j))%N);
        end
      display and store the generated test set
    end
end
```

Figure 6. Pseudocode of the proposed strategy

As a demonstration, applying the proposed strategy for N=3 results into the test set given in Figure 7.

Tuples	Tuple
1	000
2	012
3	021
4	111
5	120
6	102
7	222
8	201
9	210

Figure 7. Resulting optimum pairwise test set

4. Discussion and Conclusion

A straightforward optimum pairwise test strategy that generates optimal number of test set is presented. As seen in figure 7, the proposed strategy also gives optimum test set as OA strategy based on Latin square (seen in figure 5). Also, it can be observed that the strategy also gives a different set of test that also provides pairwise coverage. Unlike OA strategy, the proposed strategy also does not require any storage at all.

However, the proposed strategy is not without limitation. It is restricted to the condition when the number of parameters equal to number of variables, and both are primes numbers. As a scope of future work, we are currently implementing a generalized version of the strategy to support variable number of parameters.

5. References

[1] D. R. Kuhn, D. R. Wallace, A. M. Gallo, "Software Fault Interactions and Implications for Software Testing", *IEEE Transaction on Software Engineering*, 30(6), IEEE Computer Society, 2004, pp. 418-421.

[2] D. M. Cohen, S. R. Dalal, M. L. Fredman, G. C. Patton, "The AETG system: An Approach to Testing Based on Combinatorial Design", *IEEE Transaction on Software Engineering*, 23 (7), 1997, pp. 437-443.

[3] K. A. Bush, "Orthogonal arrays of index unity", Annals of Mathematical Statistics, 23 (1952), pp. 426-434.

[4] T. Shiba, T. Tsuchiya, T. Kikuno, "Using Artificial Life Techniques to Generate Test Cases for Combinatorial Testing". *Proceedings of the 28th Annual International Computer Software and Applications Conference (COMPSAC'04)*, Hong Kong, China, September 2004, pp. 72-77.

[5] Y. Lei and K. C. Tai, “In-Parameter-Order: A Test Generating Strategy for Pairwise Testing”, *Proceedings of 3rd IEEE Intl. Symp. on High Assurance System Engineering*, Nov. 1998, pp. 254-261.

[6] Y. Lei and K. C. Tai, “In-Parameter-Order: A Test Generating Strategy for Pairwise Testing”, *IEEE Transaction on Software Engineering*, 28(1), 2002, pp. 1-3.

[7] J. Yan, J. Zhang, “Backtracking Algorithms and Search Heuristics to Generate Test Suites for Combinatorial Testing”, *Proceedings of the 30th Annual International Computer Software and Applications Conference (COMPSAC'06)*, Chicago USA, IEEE Computer Society, September 2006, pp. 385 – 394

[8] M. I. Younis, K. Z. Zamli and N.A.M. Isa, “IRPS - An Efficient Test Data Generation Strategy for Pairwise Testing”, accepted for publication in KES 2008.

[9] R. Krishnan, S. M. Krishna and P. S. Nandhan, “Combinatorial Testing: Learnings from our Experience”, *ACM SIGSOFT Software Engineering Notes*, 32(3), May 2007, pp. 1-8.

[10] R. Mandl, “Orthogonal Latin squares: an application of experiment design to compiler testing”, *Communications of the ACM*, 28(10), Oct. 1985, pp. 1054-1058.

[11] A. W. Williams and R. L. Probert, R.L., “A Practical Strategy for Testing Pair-Wise Coverage of Network Interfaces”. *Proceedings of the 7th International Symposium on Software Reliability Engineering (ISSRE)*, White Plains, New York (1996).

[12] M. Grindal, J. Offutt and S. F. Andler, “Combination Testing Strategies: A Survey, GMU Technical Report ISE-TR-04-05, July 2004.

[13] http://www.research.att.com/~njas/oadir.