VOL. 10, NO. 18, OCTOBER 2015 ISSN 1819-6608
ARPN Journal of Engineering and Applied Sciences

www.arpnjournals.com

SURVEY ON INPUT OUTPUT RELATION BASED COMBINATION TEST DATA GENERATION STRATEGIES

AbdulRahman A. Alsewari[1], Nasser M. Tairan[2] and Kamal Z. Zamli[1]
[1]Software Engineering Research Group, Faculty of Computer Systems and Software Engineering, Universiti Malaysia Pahang (UMP), Kuantan, Pahang, Malaysia
[2]College of Computer Science, King Khalid University, Saudi Arabia
E-Mail: alsewari@ump.edu.my

ABSTRACT

Combinatorial test data generation strategies have been known to be effective to detect the fault in the product due to the interaction between the product's features. Over the years, many combinatorial test data generation strategies have been developed supporting uniform and variable strength interactions. Although useful, these existing strategies are lacking the support for Input Output Relations (IOR). In fact, there are only a handful of existing strategies addresses IOR. This paper will review the existing combinatorial test data generation strategies supporting the IOR features specifically taking the nature inspired algorithm as the main basis. Benchmarking results illustrate the comparative performance of existing nature inspired algorithm based strategies supporting IOR.

Keywords: combinatorial testing, test data generation, combinatorial optimization problem, nature based algorithms, software testing.

INTRODUCTION

In recent decades, researchers have developed combinatorial test data generation strategies based on different approaches in the literature to solve the combinatorial optimization problem of test suite generation (Nie, Wu *et al.* 2015; Palacios, García-Fanjul *et al.* 2015; Pérez Lamancha, Polo *et al.* 2015). Most of the existing combinatorial test data generation strategies addresses the supporting of uniform and variable interaction strength. Although helpful in addressing the uniform interaction strength and variable interaction strength, existing strategies lackthe ability to generate test cases when the system configuration dictates specific parameter interactions (called Input Output based relations (IOR)). Some of existing strategies that addresse IOR include (Density (Bryce and Colbourn, 2007; Bryce and Colbourn, 2009), TVG (Tung and Aldiwan, 2000; Arshem 2009), Union (Schroeder, 2001), Greedy (Schroeder and Korel, 2000), Para Oreder (Ziyuan, Changhai *et al.* 2007), ReqOrder (Ziyuan, Changhai *et al.* 2007), ITTDG(Othman and Zamli 2011), and AURA(Ong and Zamli 2011)).

Recently there are some of the existing combinatorial test data generation strategies adopt nature based algorithms to construct a combinatorial test data suite including Genetic Algorithm (GA) (Shiba, Tsuchiya *et al.* 2004), Ant colony Algorithm (ACA) (Shiba, Tsuchiya *et al.* 2004), Simulated Annealing (SA)(Cohen, Gibbons *et al.* 2003), and Particle Swarm Test suite Generation (CPSO) (Jarboui, Cheikh *et al.* 2007). These nature inspired strategies are effective to support both uniform interaction strength and variable interaction strength in the sense that they generate satisfactorily optimum results in most cases. Indeed, nature based strategies have great potentials, yet their strength have not been have sufficiently been tapped to support IOR. Addressing these issues, there is need to investigate further on the adoption of the nature inspired algorithm increasing the flexibility of interaction testing through supporting IOR.

MODEL FOR IOR TEST SUITE GENERATION

To elaborate of the IOR support, we have adopted the program (Pl) from (P. J. Schroeder, P. Faherty *et al.* 2002). The model has four inputs (Inputs = {A, B, C, D} and three outputs (Outputs = {E, F, G}). The output (E) is the combinations between (A, and B), the output (F) is the combination between (A, and C) and the last combination output (G) between (B, and D). (See Figure-1).

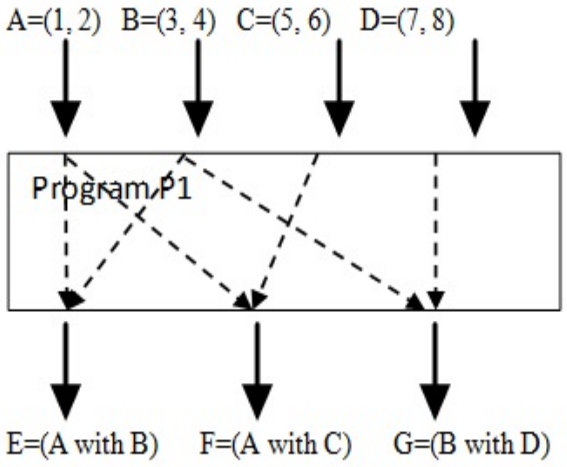

Figure-1. IOR for Program Model (P1).

As shown in Figure-1 the combination between the parameters ((A with B), (A with C), and (B with D) can be considered as (partial) pairwise combinations as not all pair combinations need to be covered. Figure-2 shows all the unique combinations whilst Table-1 illustrates the test data that have been constructed based on pairwise combinations.

Tuple ID	A	B	C	D
Tuple 1	1	3	x	x
Tuple 2	1	4	x	x
Tuple 3	2	3	x	x
Tuple 4	2	4	x	x

Tuple ID	A	B	C	D
Tuple 1	1	x	5	x
Tuple 2	1	x	6	x
Tuple 3	2	x	5	x
Tuple 4	2	x	6	x

Tuple ID	A	B	C	D
Tuple 1	1	x	x	7
Tuple 2	1	x	x	8
Tuple 3	2	x	x	7
Tuple 4	2	x	x	8

Tuple ID	A	B	C	D
Tuple 1	x	3	5	x
Tuple 2	x	3	6	x
Tuple 3	x	4	5	x
Tuple 4	x	4	6	x

Tuple ID	A	B	C	D
Tuple 1	x	3	x	7
Tuple 2	x	3	x	8
Tuple 3	x	4	x	7
Tuple 4	x	4	x	8

Tuple ID	A	B	C	D
Tuple 1	x	x	5	7
Tuple 2	x	x	5	8
Tuple 3	x	x	6	7
Tuple 4	x	x	6	8

Figure-2. The interaction pairwise combinations.

Table-1. The smallest possible test data size which covered all the combinations in Figure-2.

Test ID	A	B	C	D
Test case 1	2	3	5	7
Test case 2	1	4	6	8
Test case 3	1	3	5	8
Test case 4	2	4	6	7
Test case 5	2	4	5	7
Test case 6	1	3	6	8

Considering the IOR, we can generate each unique combination for (E = (A with B), F = (A with C), and G = (B with D)) of Program P1 test data input values that influences each output variable.

Here, each output variable is based on the corresponding input combination condition (binary number, i.e. E = (1100), F = (1010), and G = (0011)). Figure-3 demonstrates all possible combinations for each one binary. Based on these combinations Table-2 displays test data size which covered all the combinations in Figure-3.

Tuple ID	A	B	C	D
E1	1	3	x	x
E2	1	4	x	x
E3	2	3	x	x
E4	2	4	x	x

(a) Comb. A with B (1100)

Tuple ID	A	B	C	D
F1	1	x	x	7
F2	1	x	x	8
F3	2	x	x	7
F4	2	x	x	8

(b) Comb. A with C (1010)

Tuple ID	A	B	C	D
G1	x	x	5	7
G2	x	x	5	8
G3	x	x	6	7
G4	x	x	6	8

(c) Comb. B with D (0011)

Figure-3. The Required combinations based on the outputs variables.

Table-2. Test suite for Program P1 based on the outputs variables E, F, and G.

Test ID	A	B	C	D
Test case 1	2	3	5	7
Test case 2	1	4	6	8
Test case 3	1	3	5	8
Test case 4	2	4	6	7

Referring to the test data size in Table-2, the test data has been reduced to merely 4 test cases from all possible 16 test cases (i.e. considering all parameters combinations). Additionally, considering only the IOR pair combinations Table-2 has less test case as compared to Table-1 which considers all pairs combinations.

RELATED WORK

In last 10 years, many combinatorial test data generation exists. In this paper, we will review the existing strategies supporting the IOR features. Based on the adopted algorithms, there are two main approaches that

can be used to classify the combinatorial test data generation strategies: pure computational approach and nature based approach. The pure computational approach strategies greedy and iteratively process to generate the test data. The approach such as (Density (Bryce and Colbourn, 2007; Bryce and Colbourn, 2009), TVG (Tung and Aldiwan, 2000; Arshem, 2009), Union (Schroeder, 2001), Greedy (Schroeder and Korel, 2000), ITTDG(Othman and Zamli 2011), ParaOreder(Ziyuan, Changhai *et al.* 2007), ReqOrder (Ziyuan, Changhai *et al.* 2007) and AURA (Ong and Zamli, 2011)) are typical examples. Recently, significant efforts are also emerging to adopt nature based strategies in this approach. Each nature based strategy mimics the behavior of the natural like (the gene of the chromosomes, ants, swarm of birds or fish, and etc.) to produce test suite. All the existing nature based strategies as (SA, GA, ACA, PSO and HSS) (Stardom 2001; Shiba, Tsuchiya *et al.* 2004) support uniform strength. While natural based strategies generates most optimum results in most cases in uniform strength, but until now not a single one of them address IOR for test data generation.

In line with such a good prospect, this paper advocates the new combination of test data generation strategy based on nature inspired algorithms.

BENCHMARKING AND DISCUSSION

This section discusses the comparison and the discussion between the existing published strategies defined the IOR such as that Density, TVG, ReqOrder, Union, Greedy, ITTDG, and AURA. All the results for abovementioned strategies are available in the published (Ziyuan, Changhai *et al.* 2007; Ong and Zamli, 2011; Othman and Zamli, 2011). The result in Tables-3 and 4 based on the two experiments based on the benchmark defined in (Ziyuan, Changhai *et al.* 2007; Ong and Zamli, 2011; Othman and Zamli, 2011). These experiments are;

- The first experiment consider a system configuration IOR(N, 3^{10}, R)
- The second experiment consider a system configuration IOR(N, 2^3 3^3 4^3 5^1, R)

In each experiment R will start with the first 10 requested interactions and then add the next 10 and the next 10 until add 60 requested interactions in the experiments.

Where R=[{1, 2, 7, 8}, {0, 1, 2, 9}, {4, 5, 7, 8}, {0, 1, 3, 9}, {0, 3, 8}, {6, 7, 8}, {4, 9}, {1, 3, 4}, {0, 2, 6, 7}, {4, 6}, {2, 3, 4, 8}, {2, 3, 5}, {5, 6}, {0, 6, 8}, {8, 9}, {0, 5}, {1, 3, 5, 9}, {1, 6, 7, 9}, {0, 4}, {0, 2, 3}, {1, 3, 6, 9}, {2, 4, 7, 8}, {0, 2, 6, 9}, {0, 1, 7, 8}, {0, 3, 7, 9}, {3, 4, 7, 8}, {1, 5, 7, 9}, {1, 3, 6, 8}, {1, 2, 5}, {3, 4, 5, 7}, {0, 2, 7, 9}, {1, 2, 3}, {1, 2, 6}, {2, 5, 9}, {3, 6, 7}, {1, 2, 4, 7}, {2, 5, 8}, {0, 1, 6, 7}, {3, 5, 8}, {0, 1, 2, 8}, {2, 3, 9}, {1, 5, 8}, {1, 3, 5, 7}, {0, 1, 2, 7}, {2, 4, 5, 7}, {1, 4, 5}, {0, 1, 7, 9}, {0, 1, 3, 6}, {1, 4, 8}, {3, 5, 7, 9}, {0, 6, 7, 9}, {2, 6, 7, 9}, {2, 6, 8}, {2, 3, 6}, {1, 3, 7, 9}, {2, 3, 7}, {0, 2, 7, 8}, {0, 1, 6, 9}, {1, 3, 7, 8}, {0, 1, 3, 7}]

Tables-3 and 4 depict the test size results obtained for the existing IOR based strategies (Density (D), TVG (T), ReqOrder (Re), Union (U), Greedy (G), ITTDG (I), and AURA (A)) for each part. The darkened cells specify the best result for a specific test configuration.

Table-3. Comparison of test suite size generated by the existing IOR strategies for IOR (N, 310 , R).

R	D	T	Re	P	U	G	I	A
10	86	86	153	105	503	104	**81**	89
20	95	105	148	103	858	110	**94**	99
30	116	125	151	117	1599	122	**114**	132
40	126	135	160	**120**	2057	134	122	139
50	135	139	169	148	2635	138	**131**	147
60	144	150	176	142	3257	143	**141**	158

Table-4. Comparison of test suite size generated by the existing IOR strategies for IOR (N, 23 33 43 51, R).

R	D	T	Re	P	U	G	I	A
10	144	144	154	144	505	**137**	144	144
20	160	161	187	161	929	**158**	160	182
30	**165**	179	207	179	1861	181	169	200
40	**165**	181	203	183	2244	183	173	207
50	**182**	194	251	200	2820	198	183	222
60	**197**	209	250	204	3587	207	199	230

Referring to the first configuration IOR(N, 3^{10}, R) in Table-3, ITTDG produces the most optimum test suite size in all cases and overcomes all the existing strategies except when R is 40, ParaOrder produce the most optimum result. Density, came in the second place in term of producing a good test size in all cases. Due to the worst result in Table-3, Union is the worst strategy in all cases. Unlike the Union strategy, the other strategies (AURA, ParaOrder, Greedy, ReqOrder, and TVG) produce a satisfactory test data size.

Considering the second configuration IOR(N, 2^3 3^3 4^3 5^1, R) in Table-4, Density produces most optimum test data size at the cases when R is (30, 40, 50, and 60). In the case Density does not generate the most optimum results, but it generates acceptable and close to the most optimum results which they have produced by the other strategies. Greedy generates the most optimum test suite size test suite at the cases of (R=10, and 20}. The rest of the existing strategies in Table-4 produces acceptable test suite size. Here, Union generates the worst test suite.

CONCLUSIONS

This paper presents the need for the combination test data generation and shows some of the existing corresponding strategies. The Input Output Relation feature has been introduced and illustrated by an example.

A review of the existing combination test data strategies which addresses the supporting of Input Output Relation feature has been presented. Only pure computational based combination test data generation strategies addressed the IOR feature. Until now, no of the existing nature based combination test data generation strategy addressed this feature. Benchmarking and discussion show the existing published strategies defined the IOR such as (Density (D), TVG (T), ReqOrder (Re), Union (U), Greedy (G), ITTDG (I), and AURA (A)). As a suggestion for future work, it seem useful to investigate the adoption of any nature based algorithm such (GA, PSO, or any one) for providing IOR support.

ACKNOWLEDGEMENTS

This research is partially funded by UMP RDU Short Term Grant: Development of a Pairwise Testing Tool with Constraint and Seeding Support Based on an Optimization Algorithm, and FRGS Grant: Input Output Relations Harmony Search T-way Testing Strategy.

REFERENCES

[1] Arshem J. 2009. TVG. Retrieved 20 July 2011, from http://sourceforge.net/projects/tvg.

[2] Bryce R. C. and C. J. Colbourn 2007. The Density Algorithm for Pairwise Interaction Testing: Research Articles. Software Testing, Verification & Reliability, Vol. 17, No. 3, pp. 159-182.

[3] Bryce R. C. and Colbourn C. J. 2009. A Density-Based Greedy Algorithm for Higher Strength Covering Arrays. Software Testing, Verification & Reliability, Vol. 19, No. 1, pp. 37-53.

[4] Cohen M. B., Gibbons P. B. *et al.* 2003. Constructing Test Suites for Interaction Testing. Proceedings of the 25th International Conference on Software Engineering, Portland, Oregon USA, IEEE Computer Society.

[5] Jarboui B., Cheikh M. e*t al.* 2007. Combinatorial particle swarm optimization (CPSO) for partitional clustering problem. Applied Mathematics and Computation Vol. 192, No. 2, pp. 337-345.

[6] Nie C., Wu H. *et al.* 2015. Combinatorial testing, random testing, and adaptive random testing for detecting interaction triggered failures. Information and Software Technology, Vol. 62, pp. 198-213.

[7] Ong H. Y. and Zamli K. Z. 2011. Development of Interaction Test Suite Generation Strategy with Input-Output Mapping Supports. Scientific Research and Essays, Vol. 6, No. 16, pp. 3418-3430.

[8] Othman R. R. and Zamli K. Z. 2011. ITTDG: Integrated T-way Test Data Generation Strategy for Interaction Testing. Scientific Research and Essays, Vol. 6, No. 17, pp.3638-3648.

[9] P. J. Schroeder, P. Faherty. *et al.* 2002. Generating Expected Results for Automated Black-Box Testing. Proceedings of the 17th IEEE international conference on Automated Software Engineering, Washington, DC, IEEE Computer Society.

[10] Palacios M., J. García-Fanjul *et al.* 2015. Automatic test case generation for WS-Agreements using combinatorial testing. Computer Standards & Interfaces, Vol. 38, No. 0, pp.84-100.

[11] Pérez Lamancha B., M. Polo. *et al.* 2015. PROW: A Pairwise algorithm with constRaints, Order and Weight. Journal of Systems and Software, Vol. 99, No. 0, pp. 1-19.

[12] Schroeder P. J. 2001. Black-Box Test Reduction Using Input-Output Analysis. Ph.D. Illinois Institute of Technology.

[13] Schroeder, P. J. and B. Korel. 2000. Black-box test reduction using input-output analysis. SIGSOFT Softw. Eng. Notes Vol. 25, No. 5, 173-177.

[14] Shiba T., T. Tsuchiya *et al.* 2004. Using Artificial Life Techniques to Generate Test Cases for Combinatorial Testing. Proceedings of the 28th Annual International Computer Software and Applications Conference, IEEE Computer Society.

[15] Stardom J. 2001. Metaheuristics and The Search for Covering and Packing Array. Master of Scienc Master thesis, Simon Fraser University.

[16] Tung Y.-W. and W. S. Aldiwan 2000. Automating Test Case Generation for The New Generation of Mission Software System. Proceedings of the IEEE Aerospace Conference, IEEE Computer Society, pp. 431-437.

[17] Ziyuan W., N. Changhai *et al.* 2007. Generating combinatorial test suite for interaction relationship. Proceeding of the 4th international workshop on Software quality assurance: in conjunction with the 6th ESEC/FSE joint meeting, Dubrovnik, Croatia, ACM.