Application and Adjustment of "don't care" Values in t-way Testing Techniques for Generating an Optimal Test Suite

Aminu Aminu Muazu[1,2], Ahmad Sobri Hashim[1], and Aliza Sarlan[1]
[1] Computer & Information Sciences Department, Faculty of Science and Information Technology Universiti Teknologi PETRONAS, Malaysia
[2] Computer Sciences Department, Faculty of Natural and Applied Science Umaru Musa Yar'adua University Katsina, Nigeria
Email: aminu.aminu@umyu.edu.ng, {sobri.hashim, aliza_sarlan}@utp.edu.my

***Abstract*—Exhaustive testing is a type of testing that test all possible combination of parameter values to make sure the product is free from any possible faults and errors. However, employing exhaustive testing would be impossible due to the factors that includes time, labor, cost, and resource constraints. Thus, combinatorial t-way testing was introduced to complement exhaustive testing problem. The t-way testing generates small test list considering that each test case can cover the greatest number of interaction tuples based on the t-way coverage. Many works were done in the field of combinatorial test generation to generate an optimized test suite based on a given t-way interaction strength, since t-way testing is exceedingly complicated (NP-hard). The t-way techniques make use of "don't care" values when generating optimal test suite, but most of research adopt the values without a critical look into the consequences around them that lead to a higher number of test cases generated. As such, this paper presents the influence of "don't care" values in the field of t-way testing, we also show that this line of research is threatening to lead the area away from optimization rigor. A case study is given as our practical example, we applied each t-way techniques up to two times to show the impact of "don't care" values. Moreover, we suggest to the proper way of adopting "don't care" values for t-way testing strategies. Finally, our suggestion can be applying when implementing t-way strategy consciously adopt the adjustment to reduce test suite size.**

***Index Terms*—software testing, exhaustive testing, t-way techniques, test case, covering array notation, don't care values**

I. Introduction

Nowadays, people tend to rely too much on computer software, in fact it has become a segment of our lives as we cannot live without software, whether the software is mobile application, TV application, computer application, and so on. Also, in the business we carry out each day with credit/debit card acquisitions, money transfer, use of internet, e-mail, chatting, and so on we use software [1]. Update on Apple iOS 6 in 2012, where the company make decision regarding them google map platform which causes failure in their mobile computing movement. This is the least usable piece of software that Apple has ever release [2]. In the currently development of software project, it is reported that around 50% of time goes to software testing [2]. However, to solve this problem, researchers are focusing on how to find a better way for testing software, cost-effective and to find debugging techniques to confirm high quality of released software product.

Combinatorial testing is a black box testing method whereby more than one combination of input parameters is used in handling testing of a software configuration system. Combinatorial testing is a specification-based testing principle that requires for each t-way (t is the interaction strength) combination of input parameters of a configuration system, for every combination of a valid parameter value can be covered by at least one test case [2]-[4]. The combinatorial testing explores interactions among parameters with a little number of test cases. This approach has proven success in providing a strong and low-cost testing in the real-world situations.

In pairwise testing, the issues that is considered most for researchers is that each pairwise interaction most be covered by at least one test in the final test suite. Empirical result indicates that pairwise testing is practical and effective in various configuration systems. Most of the existing research on t-way testing focus on pairwise testing, which the aim of detecting fault caused when parameters interact. However, this fault is caused when two or more parameters interact. But the consideration of higher interaction strengths (t >=3) can be challenging. When the parameter interaction coverage t increases to more than 2, the number of test cases also increases exponentially [5]. It is apparent for a complex configuration system with many parameters and considering a higher t-way interaction strength (t) can lead to a combinatorial explosion problem since it is considered as NP-hard [5].

NP-hard and NP-complete problems are class of problems in computational methods that can be solved by a nondeterministic machine in polynomial time [6]. In

Manuscript received December 14, 2021; revised May 9, 2022.

doi: 10.12720/jait.13.4.347-357

prominent literature [7], it is stated that most of the hard problems are tightly related and can be translated into one another in polynomial deterministic time, this translation is known as polynomial reduction. A problem that is at least as hard as any other problem is called NP-hard whilst NP-hard problems that are contained in another NP are called NP-complete. Example of NP-hard problems are Traveling Salesman Problem [8], [9], Shortest Path Problem [10], Knapsack Problem [11], t-way testing [12], [13], and so on.

In combinatorial testing, test cases are generated by combining values from each parameter which will reduce the systems failure and increase the dependability of a system. Normally, the combinatorial testing is based on t-way combination of parameter values of a configuration system, and every combination can be covered by at least one test case [14]. The t-way testing is a sampling strategy that minimized combinatorial test data of a configuration system, and it is achieved based on a given interaction strength (t). As a result, researchers of these days are working on a sampling strategy that support an interaction testing. The past studies [15], [16] reported that t-way strategies are developed to support the following combinatorial t-way interaction: uniform, variable or input-output relation. Moreover, its reported that t-way testing strategies are generalized to support two main approaches that is either One-Parameter-at-a-Time (OPAT) or One-Test-at-a-Time (OTAT). Even though, there is no single strategy that can claim it always generate best test suite size for any configuration system, since t-way testing falls under NP-hard problem [17], [18]. However, these strategies adopt t-way techniques and make use of "don't care" values when generating optimal test suite, but most of them use the "don't care" values without a critical look into the consequences around them that lead to a higher number of test cases generated. Thus, the finding in this paper will add more adjustment to the proper way of using "don't care" value at the time implementing t-way testing strategy.

The rest of this paper is organized as follows. In Section II, we give the definitions of mathematical covering array notations. Then Section III discusses the related works on t-way interaction techniques. In addition, Section IV present the methodology using a case study. Then Section V present result of the research finding. Then Section VI is discussion that talk about the possible adjustment for "don't care" values. Finally, Section VII present the conclusion of the study.

II. Covering Array Notation

A Covering Array notation (CA) is a mathematical notation representing a configuration system with parameters and their respective values. Currently, CAs appears as another option to exhaustive testing because it is represented with all interactions of the components which can be used as a final test suite. Let elaborates the definition of these useful notations for CA notation.

Definition 1: The Covering Array notation (CA) has four parameters; let's say: *A*, *m*, *i*, and *n*. It can be represented mathematically as: $CA\ (A, m, i^n)$ [4], [19], in which the symbols *A*, *m*, *i* and *n* represents number of test cases, interaction strength, number of values and number of parameters for a given CA configuration, respectively. For example: $CA\ (9, 2, 3^4)$ refer to a configuration system's final test suite, which consist of 9 test cases, that cover 2-way interaction, and 4 parameters with 3 values each.

Definition 2: The Mixed Covering Array notation (MCA) has three parameters; let's say: *A*, *m*, and a configuration *U*. It can be represented mathematically as: *MCA (A, m, U)* [4], [19], the *A* and *m* carries the same meaning as in CA, MCA espouses a new symbol called *U*. *U* is a configuration in the following format: $(n_1^{i1}, n_2^{i2}, n_3^{i3}, n_4^{i4}, \ldots, n_z^{iz})$ indicating that there are *i1* parameters with n_1 values, *i2* parameters with n_2 values, *i3* parameters with n_3 values, *i4* parameters with n_4 values and so on. For example: $MCA\ (1265, 4, 10^2\ 4^1\ 3^2\ 2^7)$ refer to a configuration system's final test suite, which consist of 1265 test cases, that covers 4-way interaction, with have 12 parameters: 2 parameters with 10 values each, 1 parameter with 4 values, 2 parameters with 3 values each, and 7 parameters with 2 values each.

Definition 3: The Variable Covering Array notation (VCA) is more complicated that CA or MCA that has four parameters; *A*, *m*, a configuration *U*, and a set *G*. It can be represented mathematically as: *VCA (A, m, U, G)* [4], [19], the *A*, *m*, and *U* carries the same meaning in MCA, then *G* is a set consist of either CA or MCA with an interaction strength larger than *m*. For example: $VCA\ (12, 2, 3^2\ 2^2, \{CA\ (3, 3^1\ 2^2)\})$ refer to a configuration system's final test suite, which consist of 12 test cases, that covers 2-way interaction (as the main strength) for 2 parameters with 3 values each, and 2 parameters with 2 values each, it also covers 3-way interaction (as the sub strength) for 1 parameter with 3 values, and 2 parameters with 2 values each.

Definition 4: The Input-Output relation (IOR) adopts VCA notation which can be represented mathematically as: $IOR\ (A, \{K1, K2, \ldots, Kr\}, n_1^{i1}, n_2^{i2}, n_3^{i3}, n_4^{i4}, \ldots, n_z^{iz})$ [4], [19], the *A*, *i*, and *n* carries the same meaning in VCA, where *K* consists of more than one set of parameters relationship definition that will contribute toward the outputs, these set of parameter *K* can be indexing starting from *0, 1, 2, ..., z-1*. For example: $IOR\ (9, \{\{0,1,2\}, \{0,3\}\}, 2^4)$. Here, we are going to apply IOR for the two set of parameters to optimize the test suite which could lead to 9 test cases. The first parameter has three input data *{0, 1, 2}*, these input data mean the interaction strength is *3* and represent the index of the main parameters where *0* pointing the first parameter, *1* pointing the second parameter and *2* pointing the third parameter. The same for the second parameter *{0, 3}* that is for the 3-way interaction of the main parameter and represent the index of the main parameters where *0* pointing the first parameter, and *3* pointing the fourth parameter. The two set of function parameters are specified as *f1(A, B, C)* & *f2(A, D)* to consider the IOR.

III. RELETED WORK

The main aim of any t-way strategy is to systematically generate a small test list considering that each test case can cover the greatest number of interaction tuples based on the t-way coverage [4], [20]. However, several useful t-way strategies have been developed in the last 10 years that provide three types of interaction for test data generation as either uniform interaction strength, variable interaction strength, or input output-based relations.

The next subsections give an overview of these t-way interactions with some of their existing t-way strategies.

A. Uniform Interaction Strength

The uniform interaction strength uses the CA and MCA notations; it is the basis for interaction testing that considers one strength of interaction between the parameters. Here, t-way has the same strength interaction for all parameters values and to test all interacting parameters, the test suite must cover all the t-way interaction at least once [2], [21] For instance, if t=2, then it will involve combination of values for every two parameters. Example of t-way strategies that support uniform interaction strength are PWiseGen [22], OPAT-HS [2], GS [23], GTHS [3], TACO [17], and so on.

The OPAT-HS [2] is implemented to adopt OPAT approach using harmony search algorithm by Alsewari *et al.* in 2018. OPAT-HS strategy support small uniform t-way interaction strength ($2 \geq t \leq 3$).

The GS [23] is implemented to adopt OTAT approach using genetic algorithm by Sajad *et al.* in 2018. GS strategy support small uniform t-way interaction strength which are less or equal to 6 ($2 \geq t \leq 6$).

The TACO [17] is implemented to adopt OTAT approach based on ant colony optimization by Ahmad *et al.* in 2021. TACO strategy support small uniform t-way interaction strength up to 5 ($2 \geq t \leq 5$).

B. Variable Strength Interaction

Unlike uniform interaction strength, Variable strength interaction is applied for VCA notation with a combination of CA or MCA having more than one interaction strength. The VCA notations used in variable strength interaction most have different number of parameters between the VCA and its subset (CA or MCA). In VCA, the interaction dependency of subset parameters is higher than other parameters, and with due failure, the interaction of that subset parameter will have impact to the system at all [2], [21]. The variable interaction strength can be applied to any software configuration system or the system that need running multiple configurations [21]. Example of t-way strategies that support variable interaction strength are HSS [24], ABCVS [25], GAMIPOG [26], SCAVS [12], TTSGA [13], and so on.

The ABCVS [25] is implemented to adopt OTAT approach based on ant bee colony algorithm by Alazzawi *et al.* in 2019. ABCVS strategy support variable t-way interaction strength less or equal to 6 ($2 \geq t \leq 6$).

The SCAVS [12] is implemented to adopt OTAT approach based on sine cosine algorithm by Jalal *et al.* in 2020. SCAVS strategy support variable t-way interaction strength less or equal to 3 ($2 \geq t \leq 3$).

The TTSGA [13] is implemented to adopt OTAT approach using on ant colony algorithm by Ramli *et al.* in 2021. The TTSGA strategy support variable t-way interaction strength less or equal to 7 ($2 \geq t \leq 7$).

C. Input-Output Relation

The input-output relation IOR adopt VCA notation as: IOR (N, {μ1…. μk}), λ1ℓ1, λ2ℓ2, λ3ℓ3… λjℓj). Where μ consists of more than one set of parameters relationship definition that will contribute toward the outputs. These set of parameters μ can be indexing starting from 0, 1, 2, …, n-1. Normally, IOR consider interaction of input parameters that affect the output, since not all input parameter interacts with each other and the strength for each interaction is different [2], [21]. Example of t-way strategies that support IOR are ParaOrder [27], AURA [28], CTJ [29], IOR_HH [30] and so on.

The AURA [28] is implemented to adopt OTAT approach by Ong *et al.* in 2011. AURA strategy is not optimization-based strategy. The authors use the AURA strategy to examine the effectiveness of automated mapping on input-output relationship and its flexible control for generating test cases.

The CTJ [29] is implemented to adopt OTAT approach based on jaya algorithm by Younis *et al.* in 2020 to support input-output interaction strength. Moreover, CTJ focus on generating an acceptable optimal test suite size in a short execution time.

The IOR_HH [30] is presented recently as a new t-way strategy that support input-output interaction strength by Din F., and Zamli K., in 2021. IOR_HH strategy is based on the exponential Monte Carlo with counter (EMCQ) as hyper-heuristic that works a controller of the three low-level metaheuristic operators: crossover, peer learning and global pollination.

IV. METHODOLOGY

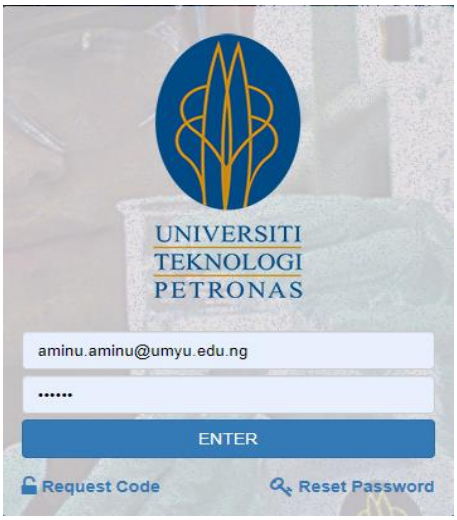

Figure 1. UTP admission login system.

In this study, a method of t-way techniques is applied to our case study configuration up to three time to

apprehend the influence of "don't care" values. We started by depicting the exhaustive testing problem and then we outline the effect of "don't care" value application in t-way techniques. To further illustrate the exhaustive testing problem, a simplified UTP Admission Login System is elaborated as shown in Fig. 1. This configuration system contains five (5) input parameters: Username, Password, ENTER, Request Code, and Reset Password. Each parameter having two possible values: Username = {Enter, Not Enter}, Password = {Enter, Not Enter}, ENTER = {Yes, No}, Request Code = {Yes, No}, and Reset Password = {Yes, No}.

The described parameters and their values in symbolic forms are displays in Table I and Table II respectively. If we test this system by exhaustive testing, it will be required value parameter as $Value^{Parameter} = 2^5 = 2*2*2*2*2 = 32$ combination of test cases. The exhaustive testing is expressed in the Table III.

TABLE I. UTP ADMISSION LOGIN SYSTEM'S PARAMETERS AND VALUES

Parameter	Username	Password	ENTER	Request Code	Reset Password
Values	Enter	Enter	Yes	Yes	Yes
	Not Enter	Not Enter	No	No	No

TABLE II. UTP ADMISSION LOGIN SYSTEM'S PARAMETERS AND VALUES (SYMBOLIC DESCRIPTION)

Parameter	U	P	E	C	R
Values	u1	p1	e1	c1	r1
	u2	p2	e2	c2	r2

TABLE III. UTP ADMISSION LOGIN SYSTEM BY EXHAUSTIVE TESTING

Test case	U	P	E	C	R
1	u1	p1	e1	c1	r1
2	u1	p1	e1	c1	r2
3	u1	p1	e1	c2	r1
4	u1	p1	e1	c2	r2
5	u1	p1	e2	c1	r1
6	u1	p1	e2	c1	r2
7	u1	p1	e2	c2	r1
8	u1	p1	e2	c2	r2
9	u1	p2	e1	c1	r1
10	u1	p2	e1	c1	r2
11	u1	p2	e1	c2	r1
12	u1	p2	e1	c2	r2
13	u1	p2	e2	c1	r1
14	u1	p2	e2	c1	r2
15	u1	p2	e2	c2	r1
16	u1	p2	e2	c2	r2
17	u2	p1	e1	c1	r1
18	u2	p1	e1	c1	r2
19	u2	p1	e1	c2	r1
20	u2	p1	e1	c2	r2
21	u2	p1	e2	c1	r1
22	u2	p1	e2	c1	r2
23	u2	p1	e2	c2	r1
24	u2	p1	e2	c2	r2
25	u2	p2	e1	c1	r1
26	u2	p2	e1	c1	r2
27	u2	p2	e1	c2	r1
28	u2	p2	e1	c2	r2
29	u2	p2	e2	c1	r1
30	u2	p2	e2	c1	r2
31	u2	p2	e2	c2	r1
32	u2	p2	e2	c2	r2

If every test case needs two (2) minutes to do, the time to complete the exhaustive testing is 64 minutes. Also, if every single test case will cost RM30, the cost to complete the exhaustive testing is RM960. Looking at this problem! It is just a simple configuration system that have only 5 parameters with 2 values each. As for a large industry that has complex software applications with thousands of parameters and values, if we consider this problem, such industry will spend much cost, labor, and time in testing their regular use software applications. Therefore, this exhaustive testing is considered as impractical, the optimization of test cases is crucial measure to do testing.

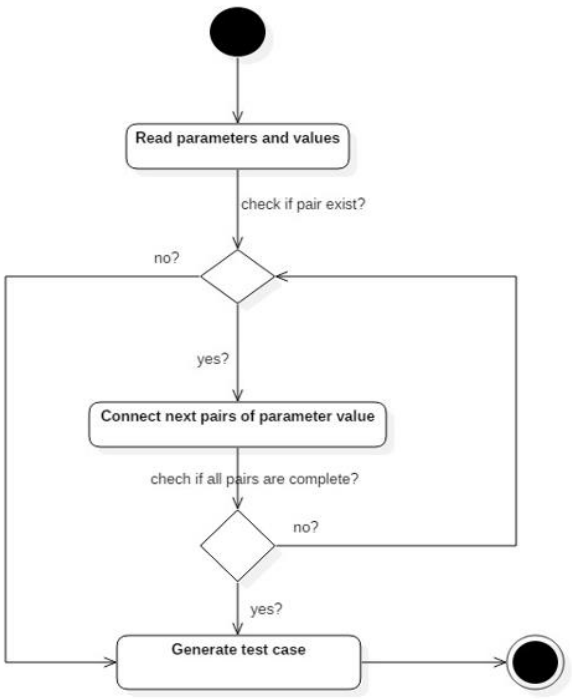

Figure 2. Generating test case.

The test case here means a set of condition which are executed with an intend of verifying the functionality of a completed software configuration system. Test cases are generated from a pair of each parameter that involved in the interaction as shown in Fig. 2. Value from each parameter is selected to form a generated test case.

Driven from this issue, the next section will consider this case study as a configuration that we applied using t-way techniques. We applied and run each t-way technique up to three times in order to show the influence of "don't care" values.

V. RESULT

To present the result, next subsections will apply t-way techniques to our case study configuration from previous section by relaxing the interaction (t) to reduce the number of test cases in the final test suite. However, we considered all the three t-way interaction strength in this case.

A. Uniform Interaction Strength

For the uniform interaction strength, let consider the interaction for t=3, then we have the following interaction for the parameters as: UPE, UPC, UPR, PEC, PER, and ECR.

VALUES	PARAMETERS				
	U	P	E	C	R
	u1	p1	e1	c1	r1
	u2	p2	e2	c2	r2
UPE Parameter Combination	u1	p1	e1	c2	r2
	u1	p1	e2	c2	r2
	u1	p2	e1	c1	r1
	u1	p2	e2	c2	r1
	u2	p1	e1	c1	r1
	u2	p1	e2	c1	r1
	u2	p2	e1	c1	r2
	u2	p2	e2	c2	r2

VALUES	PARAMETERS				
	U	P	E	C	R
	u1	p1	e1	c1	r1
	u2	p2	e2	c2	r2
PEC Parameter Combination	u1	p1	e1	c1	r1
	u2	p1	e1	c2	r1
	~~u1~~	~~p1~~	~~e2~~	~~c1~~	~~r2~~
	~~u1~~	~~p1~~	~~e2~~	~~c2~~	~~r1~~
	~~u1~~	~~p2~~	~~e1~~	~~c1~~	~~r1~~
	~~u2~~	~~p2~~	~~e1~~	~~c2~~	~~r2~~
	u2	p2	e2	c1	r2
	~~u1~~	~~p2~~	~~e2~~	~~c2~~	~~r2~~

VALUES	PARAMETERS				
	U	P	E	C	R
	u1	p1	e1	c1	r1
	u2	p2	e2	c2	r2
UPC Parameter Combination	u1	p1	e2	c1	r2
	u1	p1	e2	c2	r1
	u1	p2	e2	c1	r2
	u1	p2	e2	c2	r2
	~~u2~~	~~p1~~	~~e1~~	~~e1~~	~~r1~~
	u2	p1	e2	c2	r1
	u2	p2	e1	c1	r1
	u2	p2	e1	c2	r2

VALUES	PARAMETERS				
	U	P	E	C	R
	u1	p1	e1	c1	r1
	u2	p2	e2	c2	r2
PER Parameter Combination	~~u2~~	~~p1~~	~~e1~~	~~e2~~	~~r1~~
	u2	p1	e1	c1	r2
	~~u2~~	~~p1~~	~~e2~~	~~e1~~	~~r1~~
	~~u1~~	~~p1~~	~~e2~~	~~e1~~	~~r2~~
	~~u2~~	~~p2~~	~~e1~~	~~e1~~	~~r1~~
	u1	p2	e1	c1	r2
	u1	p2	e2	c1	r1
	~~u1~~	~~p2~~	~~e2~~	~~c2~~	~~r2~~

VALUES	PARAMETERS				
	U	P	E	C	R
	u1	p1	e1	c1	r1
	u2	p2	e2	c2	r2
UPR Parameter Combination	~~u1~~	~~p1~~	~~e1~~	~~e1~~	~~r1~~
	u1	p1	e1	c1	r2
	~~u1~~	~~p2~~	~~e1~~	~~e1~~	~~r1~~
	~~u1~~	~~p2~~	~~e1~~	~~e1~~	~~r2~~
	~~u2~~	~~p1~~	~~e1~~	~~e2~~	~~r1~~
	~~u2~~	~~p1~~	~~e1~~	~~e1~~	~~r2~~
	u2	p2	e1	c2	r1
	~~u2~~	~~p2~~	~~e2~~	~~e2~~	~~r2~~

VALUES	PARAMETERS				
	U	P	E	C	R
	u1	p1	e1	c1	r1
	u2	p2	e2	c2	r2
ECR Parameter Combination	~~u2~~	~~p2~~	~~e1~~	~~e1~~	~~r1~~
	~~u1~~	~~p1~~	~~e1~~	~~e1~~	~~r2~~
	~~u2~~	~~p2~~	~~e1~~	~~e2~~	~~r1~~
	u2	p1	e1	c2	r2
	~~u1~~	~~p2~~	~~e2~~	~~e1~~	~~r1~~
	~~u1~~	~~p2~~	~~e2~~	~~e1~~	~~r2~~
	u2	p2	e2	c2	r1
	~~u1~~	~~p2~~	~~e2~~	~~c2~~	~~r2~~

VALUES	PARAMETERS				
	U	P	E	C	R
Uniform interaction strength combination values	u1	p1	e1	c1	r1
	u2	p2	e2	c2	r2
	u1	p1	e1	c2	r2
	u1	p1	e2	c2	r2
	u1	p2	e1	c1	r1
	u1	p2	e2	c2	r1
	u2	p1	e1	c1	r1
	u2	p1	e2	c1	r1
	u2	p2	e1	c1	r2
	u2	p2	e2	c2	r2
	u1	p1	e2	c1	r2
	u1	p1	e2	c2	r1
	u1	p2	e2	c2	r2
	u2	p1	e2	c1	r2
	u2	p2	e1	c1	r1
	u2	p2	e1	c2	r2
	u1	p1	e1	c1	r2
	u2	p2	e1	c2	r1
	u1	p1	e1	c1	r1
	u2	p1	e1	c2	r1
	u2	p2	e2	c1	r2
	u2	p1	e1	c1	r2
	u1	p2	e1	c1	r2
	u1	p2	e2	c1	r1
	u2	p1	e1	c2	r2
	u2	p2	e2	c2	r1

Test cases = 25

Figure 3. Uniform interaction strength combination result when t=3 (first result).

VALUES	PARAMETERS				
	U	P	E	C	R
	u1	p1	e1	c1	r1
	u2	p2	e2	c2	r2
UPE Parameter Combination	u1	p1	e1	c1	r2
	u1	p1	e2	c1	r1
	u1	p2	e1	c1	r1
	u1	p2	e2	c2	r2
	u2	p1	e1	c2	r2
	u2	p1	e2	c2	r1
	u2	p2	e1	c1	r2
	u2	p2	e2	c1	r1

VALUES	PARAMETERS				
	U	P	E	C	R
	u1	p1	e1	c1	r1
	u2	p2	e2	c2	r2
PEC Parameter Combination	~~u1~~	~~p1~~	~~e1~~	~~c1~~	~~r2~~
	u1	p1	e1	c2	r1
	u2	p1	e2	c1	r2
	u2	p1	e2	c2	r2
	~~u1~~	~~p2~~	~~e1~~	~~c1~~	~~r1~~
	u2	p2	e1	c2	r2
	u1	p2	e2	c1	r1
	u2	p2	e2	c2	r2

VALUES	PARAMETERS				
	U	P	E	C	R
	u1	p1	e1	c1	r1
	u2	p2	e2	c2	r2
UPC Parameter Combination	u1	p1	e1	c1	r1
	~~u1~~	~~p1~~	~~e1~~	~~e2~~	~~r1~~
	u1	p2	e2	c1	r2
	u1	p2	e2	c2	r1
	u2	p1	e2	c1	r1
	u2	p1	e1	c2	r1
	~~u2~~	~~p2~~	~~e1~~	~~e1~~	~~r2~~
	~~u2~~	~~p2~~	~~e2~~	~~e2~~	~~r2~~

VALUES	PARAMETERS				
	U	P	E	C	R
	u1	p1	e1	c1	r1
	u2	p2	e2	c2	r2
PER Parameter Combination	u2	p1	e1	c1	r1
	~~u1~~	~~p1~~	~~e1~~	~~c1~~	~~r2~~
	u1	p1	e2	c2	r1
	~~u2~~	~~p1~~	~~e2~~	~~c1~~	~~r2~~
	u2	p2	e1	c2	r1
	u1	p2	e1	c1	r2
	u2	p2	e2	c1	r2
	~~u1~~	~~p2~~	~~e2~~	~~c2~~	~~r2~~

VALUES	PARAMETERS				
	U	P	E	C	R
	u1	p1	e1	c1	r1
	u2	p2	e2	c2	r2
UPR Parameter Combination	~~u1~~	~~p1~~	~~e1~~	~~c2~~	~~r1~~
	u1	p1	e2	c1	r2
	~~u1~~	~~p2~~	~~e2~~	~~c2~~	~~r1~~
	u1	p2	e1	c2	r2
	~~u2~~	~~p1~~	~~e2~~	~~c1~~	~~r2~~
	~~u2~~	~~p1~~	~~e2~~	~~e1~~	~~r1~~
	~~u2~~	~~p2~~	~~e1~~	~~e2~~	~~r1~~
	~~u2~~	~~p2~~	~~e1~~	~~c1~~	~~r2~~

VALUES	PARAMETERS				
	U	P	E	C	R
	u1	p1	e1	c1	r1
	u2	p2	e2	c2	r2
ECR Parameter Combination	~~u1~~	~~p1~~	~~e1~~	~~c1~~	~~r1~~
	~~u2~~	~~p2~~	~~e1~~	~~c1~~	~~r2~~
	~~u2~~	~~p1~~	~~e1~~	~~e2~~	~~r1~~
	u1	p1	e1	c2	r2
	~~u1~~	~~p2~~	~~e2~~	~~c1~~	~~r1~~
	~~u2~~	~~p1~~	~~e2~~	~~e1~~	~~r2~~
	~~u2~~	~~p1~~	~~e2~~	~~e2~~	~~r1~~
	~~u2~~	~~p2~~	~~e2~~	~~e2~~	~~r2~~

VALUES	PARAMETERS				
	U	P	E	C	R
Uniform interaction strength combination values	u1	p1	e1	c1	r1
	u2	p2	e2	c2	r2
	u1	p1	e1	c1	r2
	u1	p1	e2	c1	r1
	u1	p2	e1	c1	r1
	u1	p2	e2	c2	r2
	u2	p1	e1	c2	r2
	u2	p1	e1	c2	r1
	u2	p1	e2	c1	r2
	u2	p1	e2	c2	r2
	u2	p2	e1	c2	r2
	u1	p2	e1	c2	r1
	u2	p2	e1	c1	r2
	u2	p2	e2	c1	r2
	u1	p1	e2	c2	r1
	u2	p2	e2	c2	r1
	u1	p1	e1	c1	r2
	u2	p2	e1	c2	r1
	u1	p1	e1	c1	r1
	u2	p1	e1	c2	r1
	u2	p2	e2	c1	r2
	u2	p1	e1	c1	r2
	u1	p2	e1	c1	r2
	u1	p2	e2	c1	r1
	u2	p1	e1	c2	r2
	u2	p2	e2	c2	r1

Test cases = 27

Figure 4. Uniform interaction strength combination result when t=3 (second result).

If combination of parameter UPE is considered, then the values for parameter C and R takes "don't care" value (meaning that any random valid values for parameter C and R will be sufficient). Also, if combination of parameter UPC is considered, then the values for parameter E and R takes "don't care" value. Also, if combination of parameter UPR is considered, then the values for parameter E and C takes "don't care" value. Also, if combination of parameter PEC is considered, then the values for parameter U and R takes "don't care" value. Also, if combination of parameter PER is considered, then the values for parameter U and C takes "don't care" value. Finally, if combination of parameter ECR is considered, then the values for parameter U and P takes "don't care" value.

By considering these combination of parameter results, we can note that there are few reiterations of values between some entries from UPE, UPC, UPR, PEC, PER, and ECR. If we removed these repetitions, we could get all the combinations at t=3. Fig. 3 displays how the reduction is achieved. To see the effect of "don't care" values with different result obtained, let shuffle the values from each of the combination for the parameters as: UPE, UPC, UPR, PEC, PER, and ECR. Fig. 4 displays the second result after shuffling "don't care" values.

B. Variable Interaction Strength

For the variable interaction strength, let consider the mathematical notation for this application as VCA (A, 2, 2^5 (3, 2^3)), we started with the interaction strength in the main strength (for t=2). At this point, when combination of parameter UP is considered, then the values for parameter E, C, and R takes "don't care" value. Also, when combination of parameter UE is considered, then the values for parameter P, C, and R takes "don't care" value.

Also, when combination of parameter UC is considered, then the values for parameter P, E, and R takes "don't care" value. Also, when combination of parameter UR is considered, then the values for parameter P, E, and C takes "don't care" value. Also, when combination of parameter PE is considered, then the values for parameter U, C, and R takes "don't care" value. Also, when combination of parameter PC is considered, then the values for parameter U, E, and R takes "don't care" value. Also, when combination of parameter PR is considered, then the values for parameter U, E, and C takes "don't care" value. Also, when combination of parameter EC is considered, then the values for parameter U, P, and R takes "don't care" value. Also, when combination of parameter ER is considered, then the values for parameter U, P, and C takes "don't care" value. Finally, when combination of parameter CR is considered, then the values for parameter U, P, and E takes "don't care" value.

Now that we assume all combination of parameter for the interaction t=2 (main strength) for all parameters. By considering these combination of parameter results, we can note that there are few reiterations of values between some entries from UP, UE, UC, UR, PE, PC, PR, EC, ER, and CR. If we removed these repetitions, Fig. 5 could get all the combinations at t=2 for the main strength and by considering the sub strength for t=3 (here only combination of parameters ECR will be consider).

If we combine all the interactions (main and sub strength) we have the result as shown in Fig. 6. At this time, we can see that the test suite is reduced from 32 test cases of the exhaustive testing to 27 test cases. Fig. 7 show the effect of "don't care" values with different result obtained after shuffling the "don't care" values from each of the combinations (main and sub strength), while Fig. 8 display the result after shuffling "don't care" values.

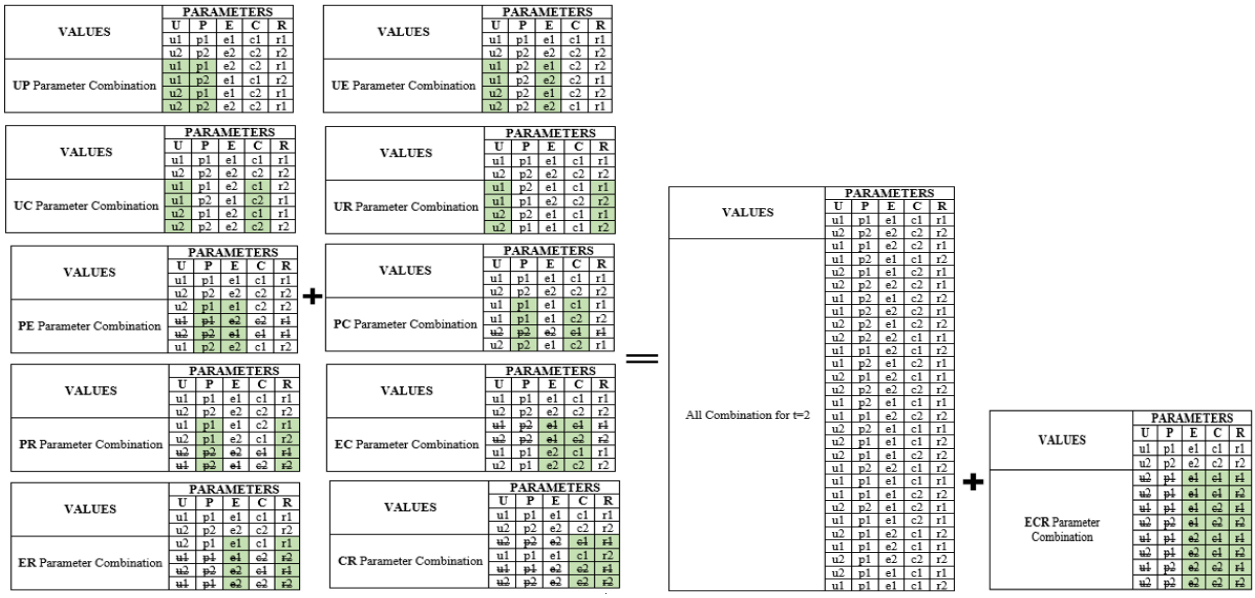

Figure 5. The VCA (A, 2, 2^5 (3, 2^3)) of all the combinations.

VALUES	PARAMETERS				
	U	P	E	C	R
	u1	p1	e1	c1	r1
	u2	p2	e2	c2	r2
Variable interaction strength combination values	u1	p1	e2	c2	r1
	u1	p2	e1	c1	r2
	u2	p1	e1	c2	r1
	u2	p2	e2	c2	r1
	u1	p2	e1	c2	r2
	u1	p2	e2	c2	r1
	u2	p2	e1	c2	r2
	u2	p2	e2	c1	r1
	u1	p1	e2	c1	r2
	u1	p2	e1	c2	r1
	u2	p1	e2	c1	r1
	u2	p2	e2	c2	r2
	u1	p2	e1	c1	r1
	u1	p1	e2	c2	r2
	u2	p2	e1	c1	r1
	u2	p1	e1	c1	r2
	u2	p1	e1	c2	r2
	u1	p2	e2	c1	r2
	u1	p1	e1	c1	r1
	u1	p1	e1	c2	r2
	u2	p2	e1	c2	r1
	u1	p1	e1	c2	r1
	u2	p1	e2	c1	r2
	u1	p1	e2	c1	r1
	u2	p1	e2	c2	r2
	u2	p1	e1	c1	r1
	u1	p1	e1	c1	r2

Test cases =27

Figure 6. The result for VCA (A, 2, 2^5 (3, 2^3)) of all the combinations (first result).

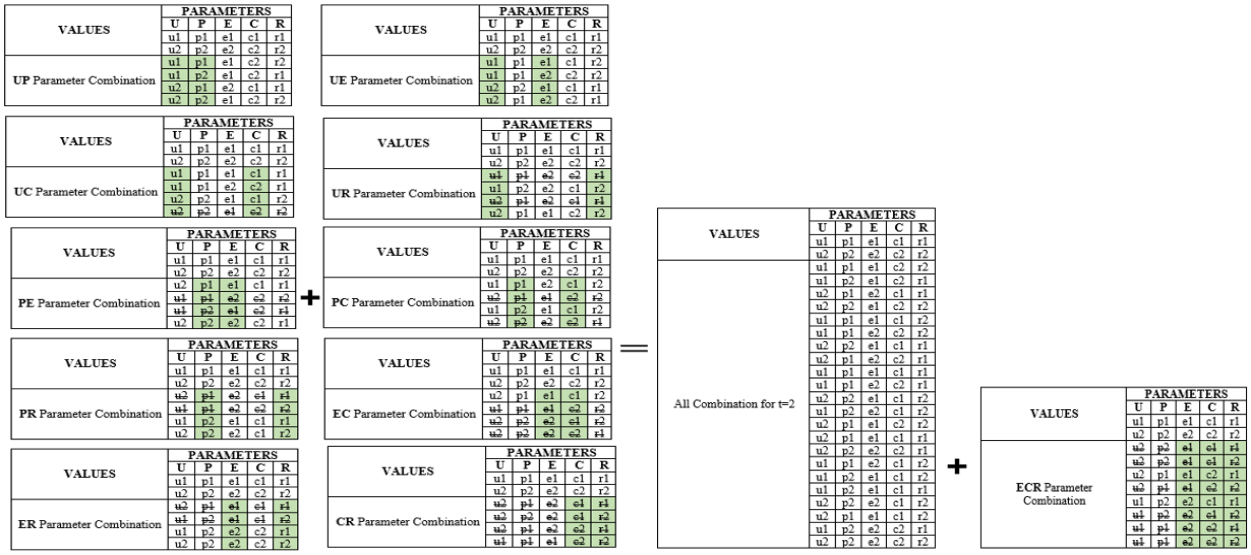

Figure 7. The VCA (A, 2, 2^5 (3, 2^3)) of all the combinations.

VALUES	PARAMETERS				
	U	P	E	C	R
	u1	p1	e1	c1	r1
	u2	p2	e2	c2	r2
Variable interaction strength combination values	u1	p1	e1	c2	r2
	u1	p2	e1	c1	r2
	u2	p1	e1	c2	r1
	u2	p2	e2	c2	r1
	u1	p2	e1	c2	r2
	u1	p2	e2	c2	r1
	u2	p2	e1	c2	r2
	u2	p2	e2	c1	r1
	u1	p1	e2	c1	r2
	u1	p2	e1	c1	r1
	u2	p1	e2	c1	r1
	u2	p2	e2	c2	r2
	u1	p2	e1	c1	r1
	u1	p1	e2	c2	r2
	u2	p2	e1	c2	r1
	u1	p1	e1	c2	r1
	u2	p1	e2	c1	r2
	u1	p1	e2	c1	r1
	u2	p2	e2	c2	r2
	u2	p1	e1	c1	r1
	u1	p2	e2	c1	r2
	u1	p1	e1	c1	r1
	u1	p1	e1	c2	r2
	u2	p2	e1	c2	r1
	u1	p1	e1	c2	r1
	u2	p1	e2	c1	r2
	u1	p1	e2	c1	r1
	u2	p2	e2	c2	r2
	u2	p1	e1	c1	r1
	u1	p2	e1	c1	r2

Test cases =24

Figure 8. The result for VCA (A, 2, 2^5 (3, 2^3)) of all the combinations (second result).

C. Input-Output Relation

For the input-output relation, let consider the mathematical notation for this application as IOR (A, {0, 3}, {0, 4}, {1, 2, 3} 2^5). The three functions from the parameters (U, P, E, C, and R) are f1, f2 and f3 in the following interactions UC, UR, and PEC respectively, that is f1 = f(UC), f2= f(UR) while f3 = f(PEC), we then

start with the interaction for f1 then f2 and then f3 accordingly.

At this point, when f1 is considered, then the values for parameter P, E, and R takes "don't care" value. Also, when f2 is considered, then the values for parameter P, E, and C takes "don't care" value. Finally, when f3 is considered, then the values for parameter U and R takes "don't care" value. By considering all these three functions (f1, f2 and f3), we are able to note that there are few reiterations of values between some entries. If we removed these repetitions, we could get all the combinations for the IOR (A, {0, 3}, {0, 4}, {1, 2, 3} 25). Fig. 9 displays how the reduction is achieved. Fig. 10 show the effect of "don't care" values with different result obtained after shuffling the "don't care" values from each of the combinations (f1, f2 and f3).

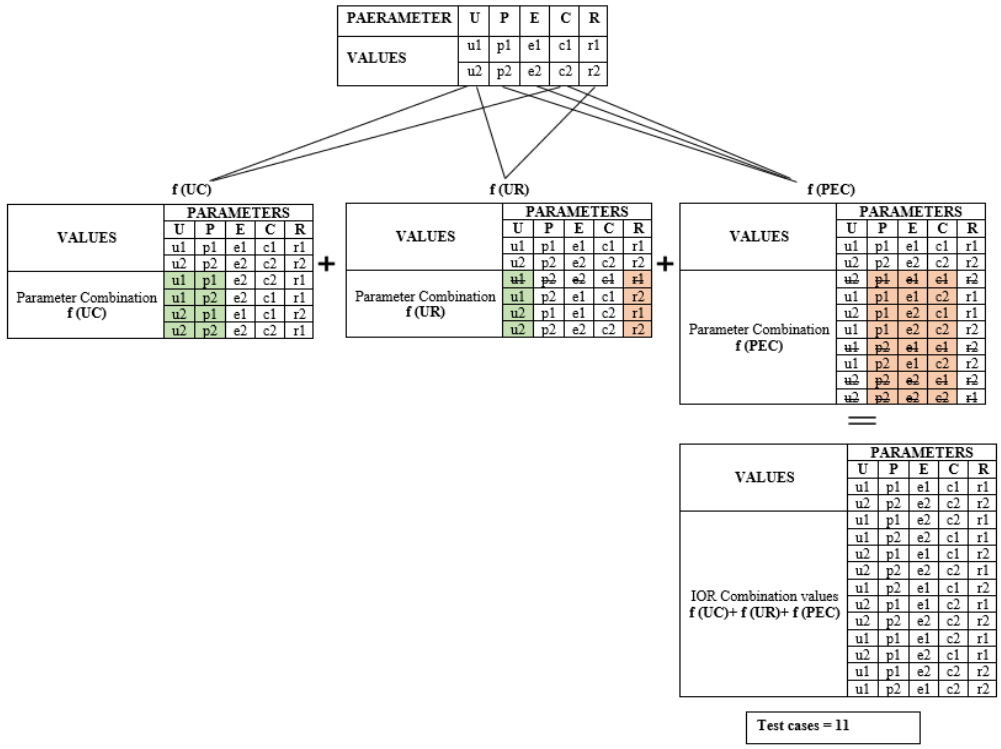

Figure 9. The IOR (A, {0, 3}, {0, 4}, {1, 2, 3} 2^5) of all the combinations result (first result).

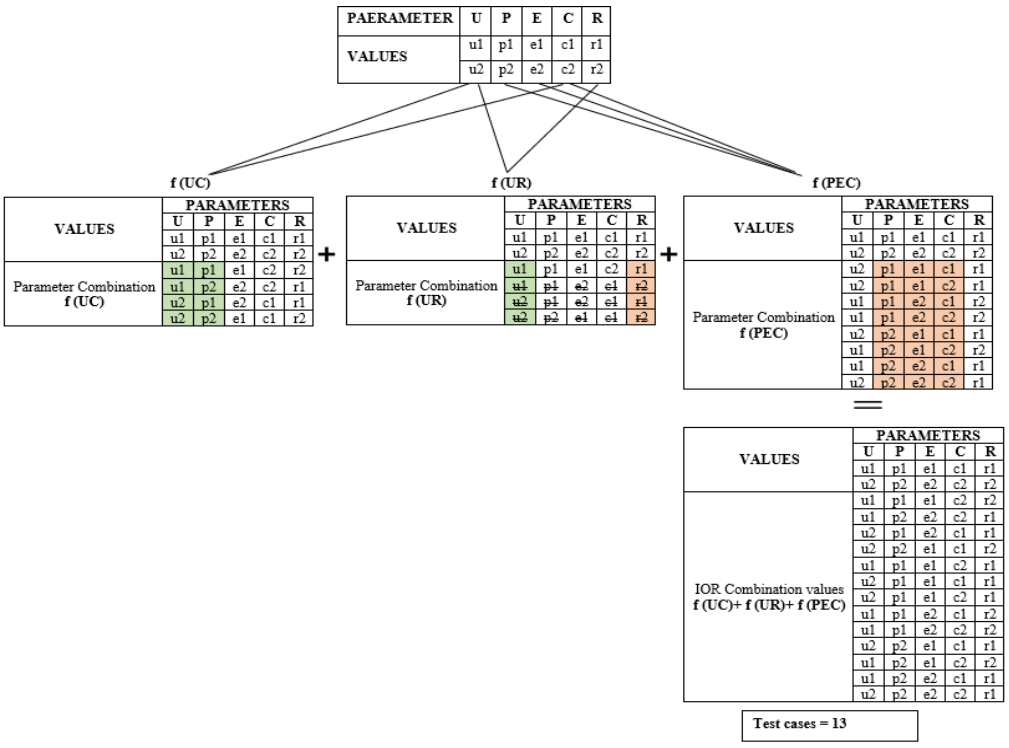

Figure 10. The IOR (A, {0, 3}, {0, 4}, {1, 2, 3} 2^5) of all the combinations result (second result).

VI. Discussion

In this section, we discussed the result obtained from previous section to demonstrate the effect of "don't care" value in t-way techniques while constructing test set for optimization. This phenomenon can be explained as we prove it by considering one configuration system as a case study. However, we run each t-way interaction up to two times. From each run, we shuffled the values of "don't care" and we noted that the result is unalike. Here comes to ask yourself why the result not the same despite we called the value as "don't care"? The Table IV depict the summary of the two different results obtained from each run.

TABLE IV. Effect of "don't care" Values

T-way	Results (test cases)	
	First	Second
Uniform interaction strength	25	27
Variable interaction strength	27	24
Input-Output Relation	11	13

In the first place, we applied the uniform interaction strength to demonstrate the influence of "don't care" values. From our first result, we obtained 25 test cases in the final test suite (see Fig. 3), then we shuffled the "don't care" values and obtained different result with 27 test cases (see Fig. 4). With these results as in Fig. 11 indicate that more attention should be given to "don't care" values, since the results are not the same.

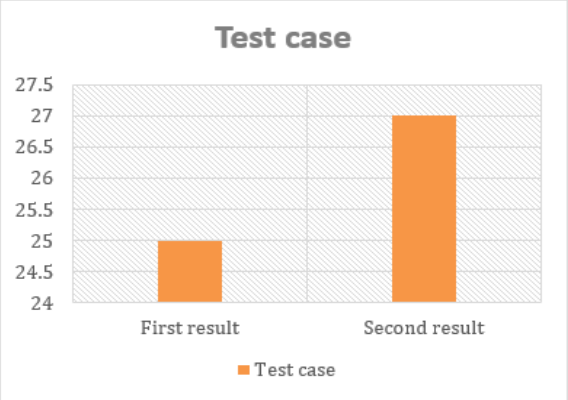

Figure 11. Uniform interaction strength results.

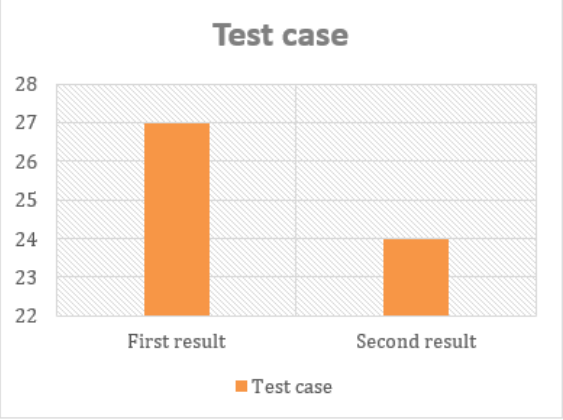

Figure 12. Variable interaction strength results.

In the second place, we applied the variable interaction strength to demonstrate the influence of "don't care" values. We obtained 27 test cases in the final test suite (see Fig. 5 and Fig. 6), then we reshuffled the "don't care" values and obtained different result with 24 test cases in the final test suite (see Fig. 7 and Fig. 8). Once more, with these results as in Fig. 12 indicate that more attention should be given to "don't care" values, since the results are not the same.

Finally, the IOR techniques is demonstrated by assuming three functions for the parameter interaction. From our first result, we obtained 11 test cases in the final test suite (see Fig. 9), then we shuffled the "don't care" values and obtained 13 test cases in the final test suite (see Fig. 10). Similarly, with these results as in Fig. 13 indicate that more attention should be given to "don't care" values, since the results are not the same.

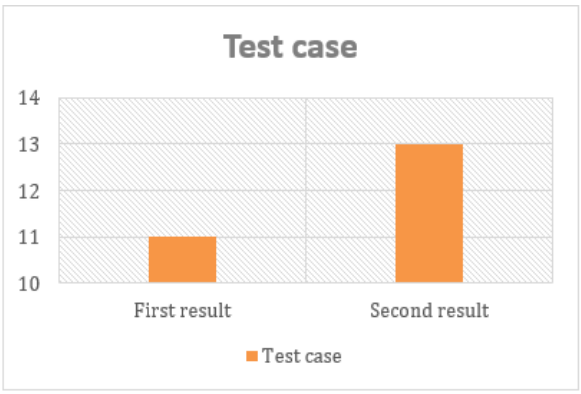

Figure 13. Input-Output relation results.

Driven from this critical issue, the finding in this research can be used to make an adjustment of using the "don't care" value while constructing test cases, particularly during implementation of any t-way strategies. For instance, at the present time of t-way strategy implementation, the authors are encouraged to run the t-way techniques several times, by shuffling the "don't care" values and allow the applied algorithm to select the best solution. Nevertheless, notwithstanding the fact that this adjustment of "don't care" values in t-way techniques will provide a better optimization in the final test suite.

VII. Conclusion

This paper presents the effect of "don't care" values in t-way interaction techniques. The appropriateness of this influence is noticeable when we run the given configuration two times for each t-way techniques. Thus, the results obtained is inconsistent to construct an optimal test suite and it is reflected in our investigational results which shows one result is better than others. However, we created a simple method of adjustment that strives to generate a better optimal test list size. To the best of author's knowledge, no paperwork has implemented a t-way strategy by overlooking into this effect of "don't care" values. The paperwork, therefore, call for a more critical look at the impact that can lead the area of t-way away from optimization rigor. Finally, a suggestion for

proper way of handling "don't care" value is created to apply during implementation of any t-way strategy. Further research direction is provided in this paper, which can drive the new researcher for future research, particularly in their research domains that are using values related to t-way's "don't care" value.

CONFLICT OF INTEREST

The authors declare no conflict of interest.

AUTHOR CONTRIBUTIONS

Conceptualization, writing—original draft preparation, methodology, resources, A. A. Muazu; validation, A. S. Hashim, and A. Sarlan; formal analysis, A. A. Muazu, and A. Sarlan; data curation, A. S. Hashim, and A. Sarlan; writing—review and editing, A. A. Muazu, A. S. Hashim, and A. Sarlan; supervision, A. S. Hashim, and A. Sarlan. All authors have read and agreed to the published version of the manuscript.

ACKNOWLEDGMENT

This research is partially funded by Universiti Teknologi PETRONAS Foundation (YUTP) with Ref. No. 015LC0-290.

REFERENCES

[1] J. Kang, S. Kwon, D. Ryu, and J. Baik, "HASPO: Harmony search-based parameter optimization for just-in-time software defect prediction in maritime software," *Applied Sciences*, vol. 11, no. 5, p. 2002, Feb. 2021.

[2] A. A. Alsewari, A. A. Mu'aza, T. H. Rassem, N. M. Tairan, H. Shah, and K. Z. Zamli, "One-Parameter-at-a-Time combinatorial testing strategy based on harmony search algorithm OPAT-HS," *Advanced Science Letters*, vol. 24, no. 10, 2018.

[3] A. R. A. Alsewari, R. Poston, K. Z. Zamli, M. Balfaqih, and K. S. Aloufi, "Combinatorial test list generation based on harmony search algorithm," *Journal of Ambient Intelligence and Humanized Computing*, pp. 1-17, 2020.

[4] A. A. Muazu and U. D. Maiwada, "PWiseHA: Application of harmony search algorithm for test suites generation using pairwise techniques," *International Journal of Computer and Information Technology*, vol. 9, no. 4, pp. 2279-0764, 2020.

[5] M. I. Younis and K. Z. Zamli, "MC-MIPOG: A parallel t-way test generation strategy for multicore systems," *ETRI Journal*, vol. 32, no. 1, pp. 73-83, Feb. 2010.

[6] R. Behmanesh, I. Rahimi, and A. H. Gandomi, "Evolutionary many-objective algorithms for combinatorial optimization problems: A comparative study," *Archives of Computational Methods in Engineering*, vol. 28, no. 2, pp. 673-688, Mar. 2021.

[7] P. Festa, "A brief introduction to exact, approximation, and heuristic algorithms for solving hard combinatorial optimization problems," in *Proc. 16th International Conference on Transparent Optical Networks*, 2014.

[8] A. H. Halim and I. Ismail, "Combinatorial optimization: Comparison of heuristic algorithms in travelling salesman problem," *Archives of Computational Methods in Engineering*, vol. 26, no. 2, 2019.

[9] Y. Saji and M. Barkatou, "A discrete bat algorithm based on Lévy flights for Euclidean traveling salesman problem," *Expert Systems with Applications*, vol. 172, 2021.

[10] F. Carrabs, R. Cerulli, and A. Raiconi, "A reduction heuristic for the all-colors shortest path problem," *RAIRO - Operations Research*, vol. 55, 2021.

[11] F. Gurski, C. Rehs, and J. Rethmann, "Knapsack problems: A parameterized point of view," *Theoretical Computer Science*, vol. 775, 2019.

[12] J. M. Altmemi, R. R. Othman, and R. Ahmad, "SCAVS: Implement sine cosine algorithm for generating variable t-way test suite," *IOP Conference Series: Materials Science and Engineering*, vol. 917, no. 1, Sep. 2020.

[13] N. Ramli, R. R. Othman, R. Hendradi, and I. Iszaidy, "T-way test suite generation strategy based on ant colony algorithm to support t-way variable strength," *Journal of Physics: Conference Series*, vol. 1755, no. 1, Mar. 2021.

[14] A. B. Nasser and K. Z. Zamli, "A new variable strength t-way strategy based on the cuckoo search algorithm," *Lecture Notes in Networks and Systems*, vol. 67, 2019.

[15] A. B. Nasser, A. A. Alsewari, A. A. Muazu, and K. Z. Zamli, "Comparative performance analysis of flower pollination algorithm and harmony search based strategies: A case study of applying interaction testing in the real world," in *Proc. 2nd International Conference on New Directions in Multidisciplinary Research & Practice*, 2016, vol. 3, pp. 51-51.

[16] A. A. Muazu and A. A. Muazu, "Design of a harmony search algorithm based on covering array t-way testing strategy," in *Proc. 1st International Conference on Information Technology in Education & Development*, 2018, pp. 33-38.

[17] M. Z. Z. Ahmad, R. R. Othman, M. S. A. R. Ali, N. Ramli, M. W. Nasrudin, and A. A. A. Halim, "A Tuned Version of Ant Colony Optimization algorithm (TACO) for uniform strength t-way test suite generator: An execution's time comparison," *Journal of Physics: Conference Series*, vol. 1962, no. 1, 2021.

[18] H. L. Zakaria, K. Z. Zamli, and F. Din, "Hybrid migrating birds optimization strategy for t-way test suite generation," *Journal of Physics: Conference Series*, vol. 1830, no. 1, 2021.

[19] K. Z. Zamli and B. S. Ahmed, "A review of covering arrays and their application to software testing," *Journal of Computer Science*, vol. 7, no. 9, pp. 1375-1385, 2011.

[20] A. A. Muazu and A. A. Muazu, "One-Parameter-at-a-Time combinatorial testing strategy based on harmony search algorithm supporting mixed covering array mathematical notation (OPATHS)," in *Proc. 1st International Conference on Information Technology in Education & Development*, 2018, pp. 64-70.

[21] N. Ramli, R. R. Othman, Z. I. A. Khalib, M. Z. Z. Ahmad, and S. S. M. Fauzi, "Ant colony algorithm to generate t-way test suite with constraints," *Journal of Physics: Conference Series*, vol. 1529, no. 4, Jun. 2020.

[22] P. Flores and Y. Cheon. (2011). PWiseGen: Generating test cases for pairwise testing using genetic algorithms. [Online]. Available: http://digitalcommons.utep.edu/cs_techrephttp://digitalcommons.utep.edu/cs_techrep/595

[23] S. Esfandyari and V. Rafe, "A tuned version of genetic algorithm for efficient test suite generation in interactive t-way testing strategy," *Information and Software Technology*, vol. 94, 2018.

[24] A. R. A. Alsewari and K. Z. Zamli, "Design and implementation of a harmony-search-based variable-strength t-way testing strategy with constraints support," *Information and Software Technology*, vol. 54, no. 6, pp. 553-568, Jun. 2012.

[25] A. K. Alazzawi, H. M. Rais, and S. Basri. (2019). ABCVS: An artificial bee colony for generating variable t-way test sets. [Online]. Available: www.ijacsa.thesai.org

[26] M. I. Younis and M. I. Younis. (2020). GAMIPOG: A deterministic genetic multi-parameter-order strategy for the generation of variable strength covering arrays. [Online]. Available: https://www.researchgate.net/publication/344599430

[27] Z. Wang, C. Nie, and B. Xu, "Generating combinatorial test suite for interaction relationship," in *Proc. Fourth International Workshop on Software Quality Assurance: In Conjunction with the 6th ESEC/FSE Joint Meeting*, 2007, p. 115.

[28] H. Y. Ong and K. Z. Zamli, "Development of interaction test suite generation strategy with input-output mapping supports," *Scientific Research and Essays*, vol. 6, no. 16, pp. 3418-3430, Aug. 2011.

[29] M. I. Younis, A. R. A. Alsewari, N. Y. Khang, and K. Z. Zamli, "CTJ: Input-Output based relation combinatorial testing strategy using Jaya algorithm," *Baghdad Science Journal*, vol. 17, no. 3, pp. 1002-1009, Sep. 2020.

[30] F. Din and K. Z. Zamli, "Hyper-Heuristic strategy for input-output-based interaction testing," *Lecture Notes in Electrical Engineering*, vol. 730, 2021.

Aminu Aminu Muazu received the B.Sc. Degree in Computer Science from Umaru Musa Yar'adua University Katsina Nigeria in the year 2011, the M.Sc. Degree in Software Engineering from Universiti Malaysia Pahang (UMP) Malaysia in the year 2017. He is currently pursuing his PhD Degree in Information Technology (Software Engineering) with the Computer and Information Science Department, Universiti Teknologi PETRONAS (UTP) Malaysia. His main research area of interest includes Software Engineering, Combinatorial t-way Software testing and Optimization Algorithm.
He is currently an academic staff holding a post as Lecturer II at Umaru Musa Yar'adua University Katsina Nigeria since 2020 to present from Computer Science department, he detained different position like Undergraduate Project Coordinator, Examination Officer and Level Advisor.
Mr. Aminu received award of best student of master's Final Year Project competition at Univerisiti Malaysia Phang in the year 2016, and Merit award of recognition as a Community Volunteer (Computer Application Instructor) at EC Computer Ltd Katsina Nigeria in the year 2014.

Ahmad Sobri Hashim is holding a post as senior lecturer at Computer & Information Sciences Department (CISD), Universiti Teknologi PETRONAS (UTP). He obtained his Doctor of Philosophy (PhD), Master of Science (MSc) and Bachelor of Technology (BTech) degrees from Universiti Teknologi PETRONAS (UTP) in Information Technology field. He has been teaching since 2014 and he is very passionate in teaching programming courses including Structured Programming and Web Programming. His research interests include IT applications, e-systems, human computer interaction, educational technologies, learning disabilities and information system. He has published quite number of research papers since 2009 and most of the papers are indexed by ISI and SCOPUS.
He is very concern on the development of society wellbeing. That is why most of the research and non-academic activities that he involved so far significantly contribute to the society. In terms of research, he has completed one research entitled A Study on Acceptance of Mobile School System in Secondary Education and Walking into Autism's Minds: Theories, Conceptual Model and Strategies for Inclusion. Currently, there are still on-going research being conducted which are Development of One-Fit-All Robot Kit for Teaching & Learning Basic Programming to Higher Education Students, Formulation of Gestural Interaction Design Model to Support Car Drivers Interacting with Smartphone User Interfaces using AHP Technique, and Formulation of Sentiment Analysis Model to Analyze HSE Situational Awareness at Oil and Gas Platform using Machine Learning. From research that he conducted, he achieved significant achievement. He won gold medal in Malaysia Technology Expo (MTE 2017), International Competition and Exhibition on Computing Innovation (ICE-CInno 2016), Pertandingan Rekacipta dan Inovasi Institusi Pengajian Tinggi Swasta (PERISTIS 2016) and International Invention, Design and Articulation (i-IDeA 2016).

Aliza Sarlan received the Bachelor. degree in IT from Universiti Utara Malaysia in 1996, Master in IT from Queensland University, Australia in 2002 and the Ph.D degree in IT from Universiti Teknologi PETRONAS (UTP), Malaysia in 2015. Her research areas of interests include human behaviour & technology adoption, information system analysis and modelling, software reliability as well as software testing. She has contributed to the areas of human factors in software development, understanding the use of technologies, and how this use can improve people's lives and their quality of life. She is now embarking into new research area of data analytics, sentiment analysis and data visualization.
Dr. Aliza is currently a researcher with the Center of Data Science (CeRDaS), UTP, where she focuses in solving complex upstream oil and gas (O&G) industry from the viewpoint computer sciences. She currently serves as the Chair of the Computer and Information Sciences Department, UTP since 2019. She is also active in conducting professional training in data analytic for public and industry. She has also been a part of the Malaysia Board of Technologist (MBOT), since 2018.