SCAVS: Implement Sine Cosine Algorithm for generating Variable t-way test suite

Jalal M Altmemi¹, R R Othman², R Ahmad³

1School of Computer and Communication, Universiti Malaysia Perlis, kampus Pauh Putra, Arau, Perlis, Malaysia,

1Iraq University College, Basra, Iraq, JalalAltmemi@gmail.com

2School of Computer and Communication, Universiti Malaysia Perlis, kampus Pauh Putra, Arau, Perlis, Malaysia,

rozmie@unimap.edu.my

3School of Computer and Communication, Universiti Malaysia Perlis, kampus Pauh Putra, Arau, Perlis, Malaysia,

badli@unimap.edu.my

corresponding author’s JalalAltmemi@gmail.com

Abstract

Exhaustive testing occurs between interaction elements (i.e., factor and level). In reality, The software system is often impractical because of other constraints, such as cost and time restrictions. The t-way testing (where t indicates the interaction strength) is one of the powerful software, and practical test techniques apply to detect interaction errors between components (i.e., produce quality minimized test cases). The t-way testing is highly complicated (NP-hard). Therefore, many Meta-heuristic algorithms are used to solve combinatorial problems successfully. In general, one of the most effective optimisation algorithm-based techniques SCA. With respect mentioned above, this paper discusses the implementation Sine Cosine Algorithm for generating a variable strength t-way test suite called SCAVS. The results of the experiments
showed that SCAVS outperformed and yielded better test sets than other existing other strategies.

Keywords: Meta-heuristic, Sine Cosine Algorithm, t-way Testing, variable-strength interaction, Test Suite Generation

1. Introduction

Meta-heuristic is a higher degree of stochastic methods to avoid local optimum by using intelligence principles of search space discovery and utilization. Meta-heuristic algorithms have been used extensively to overcome combinatorial problems that cannot be resolved in one step. Many existing works of meta-heuristic strategies have been developed including Artificial Bee Colony (ABC)(Karaboga, 2005), Cuckoo Search (CS)(Ahmed, Abdul Samad, & Potrus, 2015), Genetic Algorithm (GA)(Shiba, Tsuchiya, & Kikuno, 2004; John Stardom, 2001), Harmony Search(HS)(Alsewari & Zamli, 2012), Ant Colony Algorithm (ACA)(Shiba et al., 2004), Particle Swarm Test Generator (PSTG)(Ahmed & Zamli, 2010) Simulated Annealing (SA)(J Stardom, 2001) and Late Acceptance Hill Climbing (LAHC)(Kamal Zuhairi Zamli, 2014). However, a large number of algorithms proposed in this area, new metaheuristic algorithms are highly welcome. The term no free lunch (NFL) indicates that no single meta-heuristic algorithm can exceed others as the problem of (t − way) configurations this helps researchers to suggest new algorithms under the NFL theorem Or change new approaches to improve the existing solution.T-way testing can provide a viable solution to the interaction of components for two factors or more and producing a test set that can assist in finding fault detection early in the software(Nie & Leung, 2011; Othman, Zamli, & Mohamad, 2013). In general, not every interaction element always leads to faults; this makes t-way strategy able to achieve exhaustive testing by reducing the number of test cases.

Mathematically, the t-way test suite can be represented as a covering array. CA has four parameters used in arrays, (i.e., $CA(N; t, v, p)$ or $CA(N; t, v^p)$ ), $p_n$, number of parameters ($p_1, p_2, \ldots, p_n$ ), $v_n$ refers to the value for every single parameter ($v_1, v_2, \ldots, v_n$ ), $N$ consists of rows of test cases and $t$ represented interaction strength. Moreover, some case parameters did not interact with each other. In this case, it leads to a variable strength covering array (VSCA). VSCA can also be expressed as $VSCA(N, t, v^p(CA_1, CA_2, \ldots, CA_n))$. Here, the symbols $CA_1, \ldots, CA_n$ indicate a subset of main-set with different interaction strengths; for example, suppose we have a system with 4-parameters each parameter has 3-values where $t=3$ for main-set and $t=2$ for the sub-set. The CA can be expressed as $VCA(N, 3, 3^4, (CA(N, 2, 3^2)$). The system involves 9 test cases (rows) that are produced based on four parameters.

The rest of the paper is organized as follows. Section 2 illustrates the related work. Section 3 highlights the SCA algorithm. Section 4 SCAVS strategy parameters are tuning. Section Five and six elaborates on experimental results, and Finally, Section 7 addresses the conclusion.

2. RELATED WORK

This current section aims to present a summary of the current support research strategies that handled the support for variable strength based on AI-algorithms. Many studies have proposed a variety of t-way
approaches over the last twenty-two years. Most of these proposed strategies based on uniform t-way, (i.e., Generalized T-Way Test Suite Generator (GTWay) (Kamal Z Zamli, Klaib, Younis, Isa, & Abdullah, 2011), Artificial Bee Colony (ABC)(Ali, Othman, Yahya, Ahmad, & Ramli, 2016)), Automatic Efficient Test Generator (AETG) (David M. Cohen, Dalal, Fredman, & Patton, 1997; David M Cohen, Dalal, Kajla, & Patton, 1994), and Test Configuration (TConfig) (Williams, 2000) so on. Moreover, due to development in the software field, it is difficult to find uniformity as an interaction between parameters has become difficult. To overcome this challenge, various strategies were suggested to support the interaction of variable strengths (i.e. WHITCH(Hartman, Klinger, & Raskin, 2010), In-Parameter-Order-General IPOG (Lei, Kacker, Kuhn, Okun, & Lawrence, 2007), Parameter Ordered (ParaOrder) (Ziyuan, Changhai, & Baowen, 2007), Density(Bryce & Colbourn, 2007), Test Vector Generator (TVG)(ARSHEM, 2009), Pairwise Independent Combinatorial Testing (PICT)(Czerwonka, 2006), Simulated Annealing(SA) (M. B. Cohen, Colbourn, & Ling, 2003), Particle Swarm based Test Generator (PSTG) (Ahmed, Zamli, & Lim, 2012a, 2012b), and Integrated t-way Test Suite Generator (ITTS) (Othman, 2011)).

The previous studies have shown that a few strategies efficient and effective in producing a minimum test suite of interaction. In reality, one of the targets of this paper is to introduce a Sine Cosine Algorithm (SCA) and focus on variable interaction strength on t-way strategies.

Czerwonkais proposed PICT, a test case generation tool that depends on the t-way test strategy. It is developed to regarded the speed of test case generation, easy to use, and extensibility of the core engine facilities. The inputs (test factors/parameters and test factor level) are written in a text file format and generate any test candidate. PICT employs a greedy algorithm to search uncovered t-way tuples.

ARSHEM proposed TVG(ARSHEM, 2009) is a project-based on Multiple Document Interface (MDI) application where a screen menu item comes with submenus for moving between windows or documents. It is a type of software tester with GUI using a t-way strategy. In this strategy, the T-Reduced, Plus-One, or Random Sets algorithms are employed to generate final test cases.

Rozmie developed (Othman, 2011) algorithm Integrated t-way Test Suite Generator (ITTS). This study has provided a useful analysis and evaluation of current t-way strategies. The ITTS can support all three types of interaction (uniform strength, variable strength, or input-output based relations).

Another algorithm, ParaOrder strategy, supports VS interaction, which comes from the IPOG version. ParaOrder can facilitate prioritization of t-way test suite for its horizontal extension. In particular, the parameter with higher values will first be extended. ParaOrder supports the strength of interaction up to 3.

Density strategy was developed by Wang(Bryce & Colbourn, 2007). It can support uniform and variable strength. Density introduces minima density and maxima density calculations. A closer look reveals that density begins with an empty test case and selected one input-output relationship with the highest local density. Thus, Wang's density calculates the global density for each test case in the exhaustive interaction. Any test case with the highest global density is chosen to be moved into the Final Test Suite. This process is repeated until all interactions are covered.

Pairwise Particle Swarm Testing Generation (PSTG)(Ahmed & Zamli, 2010) strategy depends on the particle swarm optimization algorithm. It imitates the swarm behaviour of PSTG and repeatedly conducts local and global searches. PSTG starts by choosing the number of test cases that are randomly selected, and then these test cases are improved based on the velocity and the number of interactions, to find the best test cases to be added to the final test suite (FTS).
3. Sine cosine algorithm

The Sine Cosine Algorithm (SCA) is a metaheuristic algorithm recently developed based on the mathematical characteristics of the sine and cosine trigonometric functions. Mirjalili invented this algorithm in 2016 (Mirjalili, 2016). This algorithm follows the same concept of metaheuristic optimization algorithms based on the population; the first round starts with a set of random solutions; each solution upgrades its position using the below equations.

\[
X_{i}^{t+1} = X_{i}^{t} + r_{1} \sin(r_{2}) \times |r_{3}P_{i}^{t} - X_{i}^{t}| < 0.5
\]

\[
X_{i}^{t+1} = X_{i}^{t} + r_{1} \cos(r_{2}) \times |r_{3}P_{i}^{t} - X_{i}^{t}| \geq 0.5
\]

The above equations Equation 1 and Equation 2 are used by SCA depending on the value \( r_{4} \) the parameter \( r_{4} \) is a random vector which helps in the exploration and exploitation of a search space as well as in maintaining a suitable balance between them. Where \( x_{i}^{th} \) denotes the position vector of the current solution \( i^{th} \) dimension and the \( i^{th} \) iteration, \( r_{2} \) adjusts the new followed position to move local or global the best position, as shown in Figure 1 and Figure 2. \( r_{2}, r_{3} \) and \( r_{4} \) are random numbers in the range [0, 1]. The parameter \( r_{1} \) determines the radius of the search circle and altered adaptively using the following equation:

\[
r_{1} = k \frac{1 - t}{T}
\]

Equation 3
Where $k$ is a tunable constant, $t$ present current iteration and $T$ max iteration

![Figure 1](image1.png)  Direction of the current positions around the best positions  

![Figure 2](image2.png)  The range of sine and cosine functions

Thus, the pseudo code for the implement SCAVS algorithm:

**Input:** The population of solutions $X = \{X_1, X_2... X_i\}$ and the constant $k$

**Output:** The best solution $X_{best}$

1. Begin
2. Random initialization of population $X = \{X_1, X_2... X_i\}$
3. While (Not stopping criteria)
4. calculate $r_1$ using Equation 3
5. For $itr = 1$ until $itr <= T$ (max iteration)
6. For population count =1 until the count <= population size
7. Evaluate each solution within $X$ according to fitness function
8. Update the best solution obtain, $P_i^t$ = $X_{best}$
9. Generate random values for $r_2, r_3, r_4$
10. Evaluate Eq. 1 or Eq. 2 for updating the position of each solution within $X$
11. Evaluate Equation 3 for updating $r_1$ adaptively
12. End For
13. End For
14. End While
15. Obtain and display the best solution ($X_{best}$)
16. End

4. **SCAVS parameters tuning**

In order to achieve the best performance for the SCAVS strategy, the parameters need to be tuned. The performance of the SCAVS strategy largely is depended on the number of iterations and population size. To tune the control parameters, we adopt a covering array of CA ($N; 2, 4^6$) as a case study has been selected. The rationale for choosing this case study stemmed from the fact that the same case study has also been used by (Nasser, Alsewari, & Zamli, 2015). Our case study of CA ($N;2,4^6$), we vary the
Iteration with the following set of values (10,20,30,60,80,100,200,300,400, and 500) and PS from (10,40,80,100,200 and 500). Table 1 represents the smallest test suite size and its average for thirty runs of SCAVS.
Table 1 The average and best test suites for the CA \((N; 2, 4^6)\) having different values for the number PS, and iteration (Itr) was determined as follows:

| Itr | No. | Count | AVG | No. | Count | AVG | No. | Count | AVG | No. | Count | AVG | No. | Count | AVG |
|-----|-----|-------|-----|-----|-------|-----|-----|-------|-----|-----|-------|-----|-----|-------|-----|
|     | PS  |       |     | PS  |       |     | PS  |       |     | PS  |       |     | PS  |       |     |
| 10  | 23  | 2     | 25.2| 24  | 7     | 25.4| 23  | 1     | 25.4| 24  | 4     | 25.56| 22  | 1     | 25.33 |
| 20  | 24  | 9     | 25.3| 23  | 1     | 25.33| 24  | 6     | 25.33| 23  | 1     | 25.23| 24  | 6     | 25.26 |
| 30  | 22  | 1     | **25.2**| 23  | 3     | 25.1| 24  | 3     | 25.5| 23  | 1     | 25.3| 24  | 6     | 25.26 |
| 60  | 24  | 8     | 25.4| 23  | 1     | 25.53| 22  | 1     | 25.26| 23  | 1     | 25.3| 23  | 2     | 25.36 |
| 80  | 23  | 2     | 25.06| 24  | 4     | 25.26| 23  | 1     | 25.36| 24  | 4     | 25.5| 24  | 4     | 25.53 |
| 100 | 23  | 1     | 25.33| 23  | 1     | 25.06| 23  | 2     | 25.23| 23  | 2     | 25.36| 22  | 1     | 25.23 |
| 200 | 24  | 10    | 25.2| 24  | 6     | 25.33| 23  | 1     | 25.36| 23  | 1     | 25.36| 24  | 5     | 25.36 |
| 300 | 24  | 3     | 25.46| 23  | 2     | 25.43| 23  | 1     | 25.36| 23  | 1     | 25.36| 24  | 8     | 25.1  |
| 400 | 24  | 6     | 25.2| 23  | 1     | 25.23| 24  | 7     | 25.33| 24  | 6     | 25.3| 23  | 1     | 25.36 |
| 500 | 23  | 1     | 25.16| 24  | 6     | 25.36| 23  | 1     | 25.36| 23  | 2     | 25.16| 24  | 7     | 25.26 |

*Note: The values in bold are the best test suites for each Itr.*
5. Results and Discussion

To evaluate the effectiveness of the SCAVS strategy in terms of test size against other strategies, we did several experiments collected from the publications. The SCAVS strategy has initialized its parameters, such as the number of the iteration = 30 and the size of population = 10, depending on Table 1. The results obtained by the SCAVS strategy based on 30 times of running. The SCAVS strategies are implemented in Java (NetBeans IDE 8.0.2). The results are presented in Table 2 and Table 3. Each table cell shows the best result for the final test suite obtained by each strategy ((Chen, Gu, Li, & Chen, 2009),(ARSHEM, 2009),(Alsewari & Zamli, 2012),(John Stardom, 2001), and (Bryce & Colbourn, 2007)). The cells which have the dark cell represent most minimum test suite size for all strategies in that table. The table cell that is indicated (NA) presents that the result is not available. In this research, we observe from Table 2 the SCAVS and PSTG had the best results followed by ACS, and TVG respectively. Overall, the WHITCH strategy produced the worst results. With regards to Table 3 the SCAVS strategy showed a good result better than WHITCH, IPOG, PICT ACS, PSTG and TVG. SCAVS strategy produced the most minimum test set size for configuration No (5,6,7,9,12). The PICT and WHITCH strategy produce undesirable results. Whereas the results for ParaOrder, Density and ACS are not available.
Table 2 Sizes of Variable-Strength Interactional Test Suites for the Configuration TEST SUITE SIZE FOR VSCA (N, 2, 3\(^1\), ©)

| Config No | © | TEST SUITE SIZE FOR VSCA (N, 2, 3\(^1\), ©) |
|-----------|---|---------------------------------|
| WHITCH | IPOG | ParaOrder | Density | TVG | PSTG | PICT | ACS | SCAVS | SCAVS Avg. |
| 1 | Ø | 31 | 21 | 33 | 21 | 22 | 19 | 35 | 19 | 20 | 22.53 |
| 2 | CA (3, 3\(^1\)) | 48 | 27 | 27 | 28 | 27 | 27 | 81 | 27 | 27 | 29.26 |
| 3 | CA (3, 3\(^2\)) | 59 | 39 | 27 | 32 | 35 | 30 | 105 | 27 | 30 | 34.66 |
| 4 | CA (3, 3\(^3\)) | 62 | 39 | 45 | 40 | 41 | 38 | 131 | 38 | 40 | 42.4 |
| 5 | CA (4, 3\(^1\)) | 103 | 81 | NA | NA | 81 | 81 | 245 | NA | 81 | 82.93 |
| 6 | CA (4, 3\(^2\)) | 118 | 122 | NA | NA | 103 | 97 | 301 | NA | 99 | 103.43 |
| 7 | CA (4, 3\(^3\)) | 189 | 181 | NA | NA | 168 | 158 | 505 | NA | 158 | 162.26 |
| 8 | CA (5, 3\(^1\)) | 261 | 243 | NA | NA | 243 | 243 | 730 | NA | 243 | 243.8 |
| 9 | CA (5, 3\(^2\)) | 481 | 581 | NA | NA | 462 | 441 | 1356 | NA | 434 | 441 |
| 10 | CA (6, 3\(^1\)) | 745 | 729 | NA | NA | 729 | 729 | 2187 | NA | 729 | 729.36 |
| 11 | CA (6, 3\(^2\)) | 1050 | 967 | NA | NA | 1028 | 966 | 3045 | NA | 960 | 977.7 |
| 12 | CA (3, 3\(^4\)) | 61 | 53 | 49 | 46 | 48 | 45 | 146 | 45 | 45 | 48.63 |
| 13 | CA (3, 3\(^5\)) | 68 | 58 | 54 | 53 | 54 | 49 | 154 | 48 | 51 | 53.86 |
| 14 | CA (3, 3\(^6\)) | 94 | 65 | 62 | 60 | 62 | 57 | 177 | 57 | 62 | 63.33 |
| 15 | CA (3, 3\(^15\)) | 132 | NS | 82 | 70 | 81 | 74 | 83 | 76 | 81 | 83.73 |
Table 3 Sizes of Variable-Strength Interactional Test Suites for the Configuration TEST SUITE SIZE FOR VS (N, 3, 3^{15}, ©)

| Config No | © | TEST SUITE SIZE FOR SC (N, 315, ©) |
|-----------|---|-----------------------------------|
|           |   | WHITCH | IPOG | ParaOrder | Density | TVG | PSTG | PICT | ACS | SC | SC | SC | SC |
| 1         | Ø | 75     | 82   | NA        | NA      | 84  | 75   | 83   | NS  | 81 | 84 | 56 |
| 2         | CA (4, 3^4) | 129   | 87   | NA        | NA      | 93  | 91   | 1507 | NS  | 89 | 92 | 9 |
| 3         | CA (5, 3^5) | 273   | 243  | NA        | NA      | 244 | 243  | 5366 | NS  | 243 | 248 | 46 |
| 4         | CA (6, 3^6) | 759   | 729  | NA        | NA      | 729 | 729  | 12,609 | NS  | 729 | 729 | 86 |
| 5         | CA (4, 3^5) | 151   | 119  | NA        | NA      | 118 | 114  | 1793 | NS  | 111 | 116 | 16 |
| 6         | CA (5, 3^5) | 387   | 337  | NA        | NA      | 323 | 314  | 5387 | NS  | 311 | 318 | 86 |
| 7         | CA (6, 3^7) | 141   | 1215 | NA        | NA      | 1018 | 1002 | 16.792 | NS  | 961 | 976 | 43 |
| 8         | CA (4, 3^5) | 219   | 183  | NA        | NA      | 168 | 159  | 2781 | NS  | 159 | 164 | 73 |
| 9         | CA (4, 3^5) | 289   | 227  | NA        | NA      | 214 | 195  | 3095 | NS  | 203 | 206 | 33 |
| 10        | CA (4, 3^5) | 354   | 259  | NA        | NA      | 256 | 226  | 2824 | NS  | 238 | 224 | 1 |
| 11        | CA (4, 3^5) | 498   | 498  | NA        | NA      | 327 | 284  | 3632 | NS  | 306 | 309 | 2 |
| 12        | CA (5, 3^5) | 481   | 713  | NA        | NA      | 471 | 437  | 7475 | NS  | 436 | 441 | 63 |
| 13        | CA (5, 3^5) | 620   | 714  | NA        | NA      | 556 | 516  | 8690 | NS  | 526 | 531 | 33 |
| 14        | CA (6, 3^5) | 1513  | 2108 | NA        | NA      | 1479 | 1396 | 22.833 | NS  | 1396 | 1408 | 7 |
| 15        | CA (6, 3^5) | 1964  | 2124 | NA        | NA      | 1840 | 1690 | 26.729 | NS  | 1709 | 1727 | 03 |
Table 4 Post-hoc Wilcoxon Rank-Sum Tests for Table 2

| Pairs          | Ranks | Test statistics | Conclusion          |
|----------------|-------|-----------------|---------------------|
|                | SCAVS> | SCAVS< | SCAVS= | Z         | p-value in ascending order | α Holm         |                           |
| SCAVS - WHITCH | 0      | 14     | 0      | -3.29739  | 0.00098               | 0.01           | p-value < a holm Reject H0 |
| SCAVS - PICT   | 0      | 14     | 0      | -3.29577  | 0.00098               | 0.0125         | p-value < a holm Reject H0 |
| SCAVS - TVG    | 0      | 9      | 5      | -2.66791  | 0.00763               | 0.01666        | p-value < a holm Reject H0 |
| SCAVS - IPOG   | 1      | 9      | 4      | -2.65534  | 0.00792               | 0.025          | p-value < a holm Reject H0 |
| SCAVS - PSTG   | 5      | 2      | 7      | -0.17025  | 0.86481               | 0.05           | Retain null hypothesis    |
Table 5 Post-hoc Wilcoxon Rank-Sum Tests for Table 3

| Pairs       | Ranks | Test statistics | Conclusion          |
|-------------|-------|-----------------|---------------------|
| SCAVS>      | 1     | -3.23458        | p-value < α Holm    |
| SCAVS<      | 13    | 0               | Reject H0           |
| SCAVS=      | 0     | -3.17980        | p-value < α Holm    |
| WHITCH      | 1     | 0               | Reject H0           |
| TVG         | 0     | -3.17980        | p-value < α Holm    |
| PICT        | 2     | -2.91888        | p-value < α Holm    |
| IPOG        | 1     | 0               | Retain null hypothesis |
| PSTG        | 5     | 0               | Retain null hypothesis |
|            | 5     | -2.90364        | p-value < α Holm    |

6. Wilcoxon Signed-Rank Test Analysis

In order to evaluate the performance of SCAVS for t-way test generation, there is a need to statistically evaluate its relative performance (in pair) against each strategy based on the results given in both tables (Table 2 and Table 3). In this paper, the Wilcoxon test is used between SCAVS and each strategy in the experimental tables 95% confidence level (i.e. $\alpha = 0.05$). The test is performed using an SPSS software tool, where if the value is less than $\alpha$ Holm of the Asymp. Sig. (2-tailed), it indicates a
significant difference between the two sets. There are three values to evaluate ABCVS; (Ranks ABCVS>, ABCVS<, and ABCVS=) are used, as shown in Table 4 and Table 5. In other words, the results of the proposed strategy are greater, smaller, or equal to the other existing strategies. There are two values have a Statistical Test part; Asymp. Sig. (2-tailed) and Z. Regarding the Z value not considered, The value of Asymp. Sig. (2-tailed) indicates the significant difference between the two sets and that the value does not exceed $\alpha$ Holm. $\alpha$ Holm which is calculated as

$$\alpha_{Holm} = \frac{a}{m - i + 1}$$

Where $a = 0.05$, $M$ refers to the overall number of paired comparisons, and $i$ refers to the test number.

Statistical results depend on the Wilcoxon test for Table 2 and Table 3 are presented in Table 4 and Table 5. SCAVS shows there is a significant difference with other strategies in column Asymp. Sig. (2-tailed), except for PSTG, which has a significant difference from SCAVS.

7. Conclusion
Generating the most optimal variable $t$ way suite is an NP-hard problem; therefore, this field is still an active domain for research. This paper implements SCAVS into an optimization problem related to the $t$-way test generation problem. The main contribution of SCAVS supporting variable strength, SCAVS can generate a test suite up to $t=6$ and can produce a good result with suitable performance. As a scope of our future work, we are planning to enhance the SCA to support input-Out relationships as well as constraints.

References
[1] Ahmed, B. S., Abdulsamad, T. S., & Potrus, M. Y. (2015). Achievement of minimized combinatorial test suite for configuration-aware software functional testing using the cuckoo search algorithm. Information and Software Technology, 66, 13-29.
[2] Ahmed, B. S., & Zamli, K. Z. (2010). PSTG: a $t$-way strategy adopting particle swarm optimization. Paper presented at the 2010 Fourth Asia International Conference on Mathematical/Analytical Modelling and Computer Simulation.
[3] Ahmed, B. S., Zamli, K. Z., & Lim, C. P. (2012a). Application of particle swarm optimization to uniform and variable strength covering array construction. Applied Soft Computing, 12(4), 1330-1347.
[4] Ahmed, B. S., Zamli, K. Z., & Lim, C. P. (2012b). Constructing a t-way interaction test suite using the particle swarm optimization approach. *International Journal of Innovative Computing, Information and Control, 8*(1), 431-452.

[5] Ali, M. S. A. R., Othman, R. R., Yahya, Z. R., Ahmad, M. Z. Z., & Ramli, N. (2016). Implementation of artificial bee colony algorithm for T-way testing. Paper presented at the 2016 3rd International Conference on Electronic Design (ICED).

[6] Alsewari, A. R. A., & Zamli, K. Z. (2012). Design and implementation of a harmony-search-based variable-strength t-way testing strategy with constraints support. *Information and Software Technology, 54*(6), 553-568.

[7] ARSHEM, J. (2009). TVG [http://sourceforge.net/projects/tvg] [Accessed 16 Dec 2019].

[8] Bryce, R. C., & Colbourn, C. J. (2007). One-test-at-a-time heuristic search for interaction test suites. Paper presented at the Proceedings of the 9th annual conference on Genetic and evolutionary computation.

[9] Chen, X., Gu, Q., Li, A., & Chen, D. (2009). Variable strength interaction testing with an ant colony system approach. Paper presented at the 2009 16th Asia-Pacific Software Engineering Conference.

[10] Cohen, D. M., Dalal, S. R., Fredman, M. L., & Patton, G. C. (1997). The AETG system: An approach to testing based on combinatorial design. *IEEE Transactions on Software Engineering, 23*(7), 437-444.

[11] Cohen, D. M., Dalal, S. R., Kajla, A., & Patton, G. C. (1994). *The automatic efficient test generator (AETG) system*. Paper presented at the Proceedings of 1994 IEEE International Symposium on Software Reliability Engineering.

[12] Cohen, M. B., Colbourn, C. J., & Ling, A. C. (2003). Augmenting simulated annealing to build interaction test suites. Paper presented at the 14th International Symposium on Software Reliability Engineering, 2003. ISSRE 2003.

[13] Czerwonka, J. (2006). *Pairwise testing in real world*. Paper presented at the 24th Pacific Northwest Software Quality Conference.

[14] Hartman, A., Klinger, T., & Raskin, L. (2010). IBM intelligent test case handler. *Discrete Mathematics, 284*(1), 149-156.

[15] Karaboga, D. (2005). An idea based on honey bee swarm for numerical optimization. Retrieved from.

[16] Lei, Y., Kacker, R., Kuhn, D. R., Okun, V., & Lawrence, J. (2007). IPOG: A general strategy for t-way software testing. Paper presented at the 14th Annual IEEE International Conference and Workshops on the Engineering of Computer-Based Systems (ECBS’07).

[17] Mirjalili, S. (2016). SCA: a sine cosine algorithm for solving optimization problems. *Knowledge-Based Systems, 96*, 120-133.

[18] Nasser, A. B., Alsewari, A. R. A., & Zamli, K. Z. (2015). *Tuning of cuckoo search based strategy for t-way testing*. Paper presented at the International conference on electrical and electronic engineering.

[19] Nie, C., & Leung, H. (2011). A survey of combinatorial testing. *ACM Computing Surveys (CSUR), 43*(2), 1-29.

[20] Othman, R. R. (2011). DESIGN OF A T-WAY TEST SUITE GENERATION STRATEGY SUPPORTING FLEXIBLE INTERACTIONS.

[21] Othman, R. R., Zamli, K. Z., & Mohamad, S. M. S. (2013). T-way testing strategies: A critical survey and analysis. *International Journal of Digital Content Technology and its Applications, 7*(9), 222.

[22] Shiba, T., Tsuchiya, T., & Kikuno, T. (2004). *Using artificial life techniques to generate test cases for combinatorial testing*. Paper presented at the Proceedings of the 28th Annual International Computer Software and Applications Conference, 2004. COMPSAC 2004.

[23] Stardom, J. (2001). *Metaheuristics and the search for covering and packing arrays*: Simon
Fraser University Burnaby.

[26] Stardom, J. (2001). *Metaheuristics and the search for covering and packing arrays [microform]*. M. Sc. thesis, Simon Fraser University.

[27] Williams, A. W. (2000). Determination of test configurations for pair-wise interaction coverage *Testing of Communicating Systems* (pp. 59-74): Springer.

[28] Zamli, K. Z. (2014). *Generating t-way Test Suite in the Presence of Constraints*. Paper presented at the Malaysia University Conference Engineering Technology.

[29] Zamli, K. Z., Klaib, M. F., Younis, M. I., Isa, N. A. M., & Abdullah, R. (2011). Design and implementation of a t-way test data generation strategy with automated execution tool support. *Information Sciences, 181*(9), 1741-1758.

[30] Ziyuan, W., Changhai, N., & Baowen, X. (2007). *Generating combinatorial test suite for interaction relationship*. Paper presented at the Fourth international workshop on Software quality assurance: in conjunction with the 6th ESEC/FSE joint meeting.