Variable Strength Sand Cat Swarm Optimization (VSCSO) for T-Way Combinatorial Test Suite Generation

Muhammad Aiman Mohd Asyraf
Faculty of Electronic Engineering & Technology (FKTEN), Universiti Malaysia Perlis

Mohd Zamri Zahir Ahmad
Faculty of Electronic Engineering & Technology (FKTEN), Universiti Malaysia Perlis

Rozmie Razif Othman
Faculty of Electronic Engineering & Technology (FKTEN), Universiti Malaysia Perlis

Ahmad Ashraf Abdul Halim
Faculty of Electronic Engineering & Technology (FKTEN), Universiti Malaysia Perlis

他

https://doi.org/10.5109/7395602

出版情報：Proceedings of International Exchange and Innovation Conference on Engineering & Sciences (IEICES). 11, pp.783-790, 2025-10-30. International Exchange and Innovation Conference on Engineering & Sciences
バージョン：

Variable Strength Sand Cat Swarm Optimization (VSCSO) for T-Way Combinatorial Test Suite Generation

Muhammad Aiman Mohd Asyraf[1], Mohd Zamri Zahir Ahmad[1,2], Rozmie Razif Othman[1,2], Ahmad Ashraf Abdul Halim[1,2], Kentaro Go[3], Murad Muhammad Hasan Salih Al-Walidi[1], Anjila J Suali[1], Nurol Husna Che Rose[1], Fatinnabila Kamal[4]
[1]Faculty of Electronic Engineering & Technology (FKTEN), Universiti Malaysia Perlis, Malaysia
[2]Centre of Excellence for Advanced Computing (AdvComp), University Malaysia Perlis, Malaysia
[3]Department of Computer Science and Engineering, University of Yamanashi, Kofu, Japan.
[4]Institute of Engineering Mathematics, Universiti Malaysia Perlis, Malaysia
zamrizahir@unimap.edu.my

Abstract: *Generating efficient t-way combinatorial test suites remain a challenge in software testing due to the rapid growth of input combinations. This paper introduces the Variable Strength Sand Cat Swarm Optimization (VSCSO) algorithm, a metaheuristic inspired by sand cats' hunting strategies, for generating test suites with variable interaction strengths. VSCSO applies iterative search and balances exploration and exploitation to minimize the number of test cases while ensuring comprehensive coverage. Although VSCSO does not always produce the smallest test suite in every configuration, comparative experiments show it achieves competitive results and outperforms several computational methods, including TVG, IPOG, and SCIPOG-VS. Statistical analysis demonstrates that VSCSO attains competitive performance in 61.49% of evaluations, confirming its reliability and consistency. Overall, VSCSO is a promising strategy for scalable and effective combinatorial test suite generation in software testing. Future work may focus on refining VSCSO's search strategies to further enhance performance in complex scenarios.*

Keywords: t-way testing; sand cat swarm optimization; variable strength interaction testing; metaheuristic optimization; software testing.

1. INTRODUCTION

Nowadays, the rapid advancement of technology and the increasing complexity of systems designed to meet ever-evolving human needs and expectations have made it imperative for systems to function effectively and methodically [1]. Software systems' dependability is a big concern because they are widely used in many different industries, including banking, healthcare, aviation, and essential infrastructure. For instance, the application of machine learning in medical diagnostics demonstrates how high dependability is crucial in ensuring accurate, personalized healthcare interventions [2]. This emphasis on reliability is echoed across other complex systems, such as experimental platforms for nanostructure synthesis, where precise control of parameters is essential to achieve consistent and safe outcomes [3]. The standard for developed software quality assurance has increased due to the demand for software applications [4]. Software solutions must be dependable and of the highest quality to satisfy the demands and expectations of every customer [5]. In fact, the testing phase is reported to consume between 30% and 50% of the total software development cost [6], highlighting its critical role in delivering dependable software that users can trust.

As the complexity of software applications rises, so does the number of configurable parameters they encompass. This escalation results in an exponential growth in the possible input combinations. Though theoretically guaranteeing total accuracy, exhaustive testing which is a technique that tests every possible combination but at the end it becomes impracticable due to high computational costs and time constraints, as the combinatorial explosion problem renders it impracticable for a large number of input parameters [7].To solve this issue, combinatorial testing is one of the solutions that focuses on covering t-way interactions between input parameters, rather than testing every possible combination has become a successful strategy to address this problem. Compared to exhaustive testing, this method offers an organized approach that successfully lowers the number of test cases required while preserving sufficient coverage of parameter interactions [8].

T-way testing's fundamental idea is grounded in empirical findings that the majority of software errors are brought on by the interaction of a limited number of input parameters rather than the full input space, which can lead to an incorrect result [9]. According to studies, a significant portion of defects can be found using pairwise (2-way) testing, and fault detection is improved by raising the interaction strength (t). Combinatorial testing improves software quality without necessitating an unmanageable number of test cases by methodically covering all potential t-way interactions. This ensures that critical parameter interactions are tested. The choice of a suitable t determines how effective this method is; larger values result in better fault detection at the expense of a larger test suite size.

The combinatorial explosion makes it impossible to conduct exhaustive testing, which entails analyzing every possible combination of input parameters. The size of the test suite is lowered in proportion to the interaction strength, "t," and T-way testing can cover all of the significant interaction components at least once. [10]. It can reduce the expected time and cost while simultaneously increasing the efficacy of software testing from various configuration systems [11]. Combinatorial testing is also made much more efficient by it, particularly for large software systems where cost and

time are key considerations. It enables teams to avoid becoming mired in an endless cycle of exhaustive testing, identify possible problems early, optimize resources, and speed up product delivery [12].
Despite their usefulness, many of the current strategies still have trouble reaching optimality, especially when it comes to reducing the size of the test suite. Since creating a t-way test suite is an NP-hard problem, no one approach can be said to be the most effective for every testing configuration at every time [13]. Further research and innovation are made possible by this ongoing difficulty, which opens the door to the introduction of fresh approaches that can more effectively tackle these enduring problems. The design and evaluation of VSCSO, a novel t-way testing approach for variable strength interaction testing that will be put into use in 2022 [14], is presented in this study in response to these limitations. Despite showing promising results in several optimization problems, including some t-way testing applications, VSCSO has not yet been fully applied in a greater range of testing scenarios.

2. RELATED WORK

2.1 Variable Strength Covering Array

A variation on the covering array (CA) technique is called variable strength covering array (VSCA). The issue with CA is that while testing at higher strengths ensures that every parameter combination is covered at least once, it can lead to unnecessary computations and wasted resources due to redundant combinations. The covering array's limitation is that it is not able to withstand varying degrees of parameter testing, which has led to the development of VSCA. CA is initially intended to handle a single, consistent parameter value. The development of variable strength covering arrays (VSCA), which can support multiple sets of parameters and values, was prompted by this limitation [15]. It assigns varying t-way levels to parameters according to their system criticality, optimizing test coverage while avoiding unnecessary increases in test suite size[16]. VSCA represented as an equation (1),

$$S = VSCA\,(N, t, C, R) \qquad (1)$$

where C is the value of the configuration, t is the dominant interaction strength, and N is the size of the test suite. This C value can be shown as v_1^{p1}, v_2^{p2},.. v_n^{pn}, which show how the parameters and their values change. As stated in equation (1), the R is the multi-set of interacting parameters with varying strengths.

2.2 Search Technique in T-way

Research in t-way combinatorial testing has greatly progressed, shifting from computational methods to metaheuristic methods. Even though computational methods are more adaptable than algebraic methods, they remain the conventional approach. It is initially developed from specialists' knowledge and experience, to examine the search space in a notably efficient manner [13]. Computational methods, also rooted in algebraic techniques, have been fundamental in t-way test suite generation. These methods employ heuristic rules and greedy strategies to cover uncovered test combinations [17]. Nonetheless, although computational methods provide versatility and accommodate intricate setups, they face challenges in efficiency, needing more time to manage large test combinations [18]. This constraint emphasizes the necessity for more scalable solutions. Examples of computational method strategies are TVG [19], Density [20], SCIPOG-VS[21], IPOG[22], GVS[23] and DA-FO [23].

To overcome these difficulties, metaheuristic algorithms have become a viable substitute for t-way test suite generation. Usually starting with a random solution, metaheuristics use search strategies iteratively to increase fitness. Their successful use in other fields, such as ACO in IoT optimization [24] , further supports their relevance in combinatorial testing contexts. Compared to computational methods, these approaches effectively generate near-optimal solutions in less execution time, despite their imperfect accuracy. In the end, metaheuristic algorithms improve the network's capacity to resolve a variety of difficult t-way interaction problems by facilitating a more robust and flexible adaptive learning process [25]. Examples of metaheuristic method strategies are VS-TACO[13], HABCsm[26], TTSGA[17], HABC[27], GAMIPOG[28], SCAVS[29] and ABCVS[30].

2.3 Proposed Strategy of the Variable Sand Cat Swarm Optimization (VSCSO) Algorithm

The "Felis margarita" hunting and survival tactics of sand cats served as the model for the bio-inspired optimization algorithm known as VSCSO. In his thesis, Amir Seyyedabbasi and Farzad Kiani [14] originally proposed the SCSO concept as a solution to optimization difficulties. This new method effectively solves optimization problems by combining the distinctive behavioral tendencies of sand cats. Fig 1 illustrates the phases of exploration and exploitation based on sand cat hunting behavior.

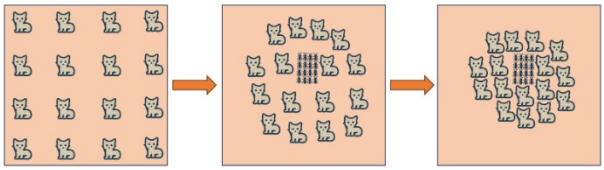

Fig 1. Exploration and Exploitation of SCSO

Inspired by sand cats' hunting habits, the VSCSO algorithm is a new metaheuristic optimization method. Each sand cat searches to locate better prey by shifting to a more advantageous position. These creatures hunt by using sound frequency to detect prey. Based on the frequency of the sound, it sets off the sand cat's hunting behavior, in which it attacks or looks for prey; each sand cat aims to catch better prey. Therefore, in order to catch better prey, the sand cat will search for a better location. The VSCSO algorithm gives the software a potent exploitation capability by having each sand cat approach its victim gradually [31] .
The framework for the VSCSO strategy is illustrated in Fig. 2, the Test Case Generator (TCG) and Tuple Generator (TG), adapted from [17],[32], are two core components developed and utilized to implement the SCSO method. The framework consists of three primary sections. The first section, Input Parameter Setting, defines the test parameters, their values, interaction

parameters, and interaction strength, forming the foundation for test case generation. The Tuple Generator (TG) applies the OTAT test case generation approach to produce a Tuple List, which contains input parameter combinations. In the second section, VSCSO Strategy, the Test Case Generator (TCG) utilizes the tuples from the Tuple List to generate optimized test cases using the SCSO algorithm. The final section produces the Final Test Suite, ensuring that all specified interactions are adequately covered.

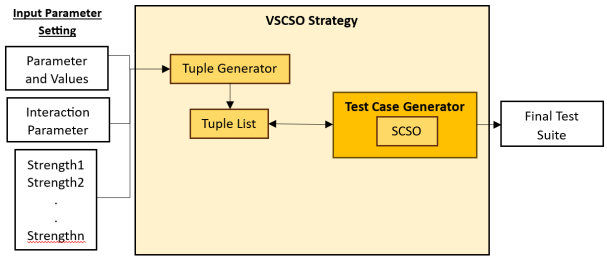

Fig 2. VSCSO's Framework

3. EVALUATION

The evaluation has been conducted to determine the effectiveness of the VSCSO by benchmarking with other existing strategies. The VSCSO algorithm is developed and compiled using the Java programming language within the Eclipse 2024 (4.34.0) software. The running environment for this evaluation of VSCSO is a desktop PC operating on Windows 11, equipped with a 2.19 GHz Intel® Core™ i7-12700F CPU and 32 GB of RAM. This project focuses solely on variable strength configurations, as VSCSO is a newly introduced approach to t-way testing. The evaluation aims to compare the best test suite size for each experiment's parameter setting. 3 main dominant test configuration groups were conducted, which are:

a) VSCA (2, 3^{15}, {S}) based on [29], [21], [27], [32], [30] and [26]
b) VSCA (3, 3^{15}, {S}) based on [29], [27], [32] and [30]
c) VSCA (N; 2, $4^3 5^3 6^2$, {S}) based on [21], [27], [32], [30] and [26].

Each table cell marked with bold numbers represented the best test suite sizes generated, while the 'NA' symbol is marked for strategy that has no available results in the published papers.

The VSCSO result published in the cells is the best result after performing all independent runs for each testing configuration, to get statistical confidence. Based on the results presented in Table 1, Table 2 and Table 3, the VSCSO strategy shows a commendable balance between optimality and consistency across different testing configurations. In Table 1, VSCSO achieves five bolded results, accounting for 31.3% optimality across 16 tests for (2, 3^{15}, {S}), indicating its competitive edge in simpler configurations.

Although its optimal rate drops in Table 2 to just one bolded result (6.7% of 15 tests for (3, 3^{15}, {S})), the differences in test suite sizes remain minimal, suggesting stable performance even in more complex scenarios. Notably, in Table 3, VSCSO performs the strongest with nine bolded results, which around 60% optimality across 15 tests that match the performance of other advanced strategies such as VS-TACO, HABCsm, TTSGA, HABC, and ABCVS for VCA (N; 2, $4^3 5^3 6^2$, {S}). When considering all tables, VSCSO can generate an optimum test suite size, which totals 15 bolded results out of 46 test cases, resulting in an overall optimality rate of approximately 32.6%. This trend highlights that while VSCSO may not always dominate in every setting, it scales well with complexity and remains highly competitive. Such results validate VSCSO as a reliable and effective approach, especially when consistent performance is preferred over occasional best-case outcomes.

Table 1. Result Test Suite Size Performance for Group

(2, 3^{15}, {S})	Metaheuristic-based strategies								Computational-based strategies					
Year	2025	2022	2021	2021	2020	2020	2020	2019	2023	2013	2012	2010	2008	2007
{S}	VSCSO	VS-TACO	HABCsm	TTSGA	HABC	GAMIPOG	SCAVS	ABCVS	SCIPOG-VS	DA-FO	GVS	TVG	Density	IPOG
Ø	22	22	**19**	22	**19**	21	20	20	NA	20	**19**	22	21	21
CA(3,3^3)	**27**	**27**	**27**	**27**	**27**	**27**	**27**	**27**	57	29	**27**	**27**	28	**27**
CA(3,3^4)	32	29	29	32	30	**27**	30	32	63	34	28	35	32	39
CA(3,3^5)	41	40	**39**	**39**	**39**	**39**	40	41	73	42	**39**	41	40	**39**
CA(3,3^6)	**45**	**45**	**45**	**45**	**45**	NA	**45**	**45**	75	46	**45**	48	46	53
CA(3,3^7)	52	49	50	49	50	NA	51	50	82	53	**48**	54	53	58
CA(3,3^9)	60	55	59	54	**50**	NA	62	58	92	60	58	62	60	65
CA(3,3^{15})	86	73	80	72	81	NA	81	81	131	78	75	81	**70**	NA
CA(4, 3^4)	**81**	**81**	**81**	**81**	**81**	**81**	**81**	**81**	110	NA	**81**	**81**	NA	**81**

CA(4, 3^5)	98	93	90	95	93	100	99	90	132	NA	99	103	NA	122
CA(4, 3^7)	156	155	153	155	154	165	158	154	173	NA	157	168	NA	181
CA(5, 3^5)	243	243	243	243	243	243	243	243	273	NA	243	243	NA	243
CA(5, 3^7)	437	435	436	428	439	461	434	446	402	NA	445	462	NA	581
CA(6, 3^6)	729	729	729	729	729	729	729	729	760	NA	729	729	NA	729
CA(6, 3^7)	957	898	830	894	830	NA	960	956	NA	NA	947	1028	NA	1196
CA(3, 3^4) **CA(3, 3^5)** **CA(3, 3^6)**	45	46	82	48	82	NA	NA	82	NA	46	42	53	46	51
Total Optimal	**5** **31.3%**	**5** **31.3%**	**10** **62.5%**	**6** **37.5%**	**9** **56.3%**	**6** **37.5%**	**5** **31.3%**	**6** **37.5%**	**1** **6.3%**	**0** **0%**	**9** **56.3%**	**4** **25%**	**1** **6.3%**	**5** **31.3%**

Table 2. Result Test Suite Size Performance for Group 2

(3, 3^{15}, {S})	**Metaheuristic-based strategies**						**Computational-based strategies**	
Year	**2025**	**2022**	**2021**	**2020**	**2020**	**2019**	**2010**	**2007**
{S}	**VSCSO**	**VS-TACO**	**TTSGA**	**HABC**	**SCAVS**	**ABCVS**	**TVG**	**IPOG**
Ø	86	80	84	85	81	81	84	82
CA(4,3^4)	95	90	93	94	89	93	93	87
CA(4,3^5)	118	112	114	115	111	115	118	119
CA(4,3^7)	159	156	152	157	159	157	168	183
CA(4,3^9)	203	194	196	199	203	196	214	227
CA(4,3^{11})	243	232	231	238	238	333	256	259
CA(4,3^{15})	322	301	308	315	306	308	327	NA
CA(5,3^5)	244	243	244	243	243	243	244	243
CA(5,3^6)	316	310	313	325	311	316	323	337
CA(5,3^7)	433	433	432	442	436	439	471	713
CA(5,3^8)	522	516	516	531	526	527	556	714
CA(6,3^6)	729	729	729	729	729	729	729	729
CA(6,3^7)	967	958	990	964	961	949	1018	1215
CA(6, 3^8)	1398	1371	1371	1414	1396	1424	1479	2108
CA(6, 3^9)	1709	1685	1683	1737	1709	1752	1840	2124
Total Optimal	**1** **6.7%**	**8** **53.3%**	**7** **46.7%**	**2** **13.3%**	**3** **20%**	**3** **20%**	**1** **6.7%**	**3** **20%**

Table 3. Result Test Suite Size Performance for Group 3

VCA (N; 2, $4^3 5^3 6^2$, {S})	Metaheuristic-based strategies							Computational-based strategies					
Year	**2025**	**2022**	**2021**	**2021**	**2020**	**2020**	**2019**	**2023**	**2013**	**2012**	**2010**	**2008**	**2007**
{S}	**VSCSO**	**VS-TACO**	**HABCsm**	**TTSGA**	**HABC**	**GAMIPOG**	**ABCVS**	**SCIPOG-VS**	**DA-FO**	**GVS**	**TVG**	**Density**	**IPOG**
Ø	45	41	42	42	42	40	44	NA	40	40	44	41	43
CA(3,4^3)	64	64	64	64	64	64	64	116	64	64	67	64	83
CA(3,5^3)	125	125	125	125	125	125	125	178	125	125	125	125	136
CA(3,$4^3 5^2$)	125	114	121	113	121	108	128	179	132	127	132	131	147
CA(3,$5^1 6^2$)	180	180	180	180	180	180	180	230	180	180	180	180	180
CA(3,$4^3 5^3 6^1$)	220	208	210	211	212	201	213	263	211	202	237	207	215
CA(3,$4^3 5^3 6^2$)	276	261	263	262	269	243	266	298	261	237	302	256	NA
CA(3,4^3) CA(3,5^3)	125	125	125	125	125	125	125	NA	125	125	125	125	136
CA(3,4^3) CA(4,$5^3 6^1$)	750	750	750	750	750	750	750	NA	NA	750	750	NA	750
CA(3,4^3) CA(5,$5^3 6^2$)	4500	4500	4500	4500	4500	NA	4500	NA	NA	4500	4500	NA	4500
CA(4,$4^3 5^1$)	320	320	320	320	320	320	320	373	NA	320	320	NA	329
CA(4,$4^3 5^2$)	469	463	461	466	461	400	463	513	NA	463	496	NA	512
CA(4,$4^3 5^1$) CA(4,$5^2 6^2$)	900	900	900	900	900	900	900	NA	NA	900	900	NA	900
CA(5,$4^3 5^2$)	1600	1600	1600	1600	1600	1600	1600	1651	NA	1600	1600	NA	1602
CA(5,$4^3 5^3$)	2429	2420	NA	2413	2411	2000	2403	2412	NA	2380	2592	NA	2763
Total Optimal	**9 60%**	**9 60%**	**9 60%**	**9 60%**	**9 60%**	**14 93.3%**	**9 60%**	**0 0%**	**5 33.3%**	**10 66.7%**	**8 53.3%**	**4 26.7%**	**4 26.7%**

Table 4 presents the results of testing conducted using the Wilcoxon and Friedman tests, involving data from Table 1, Table 2 and Table 3 with total of 13 different strategies. The performance analysis of VSCSO across 13 benchmark comparisons shows that it outperforms 23.08% of the competing strategies, specifically surpassing TVG, IPOG, and SCIPOG-VS. However, it is outperformed in 38.41% of the cases, with strategies such as VS-TACO, TTSGA, GVS, SCAVS, and HABCsm demonstrating superior results. In the remaining 38.41% of comparisons, no significant performance difference is observed, indicating that VSCSO holds its ground against methods like DA-FO, Density, GAMIPOG, ABCVS, and HABC. Overall, VSCSO proves to be competitive in 61.49% of the evaluations, demonstrating its reliability and effectiveness in combinatorial test suite generation. While its strengths are most evident against traditional computational approaches, there remains clear potential for enhancement to better compete with more advanced metaheuristic techniques. These results provide a promising foundation for future improvements, positioning VSCSO as a viable and evolving strategy that could achieve even stronger performance with further optimization and refinement.

4. CONCLUSION

The VSCSO algorithm introduces a new and efficient approach for t-way combinatorial test suite generation. It is capable of producing compact yet comprehensive test suites, thanks to its strategic balance between exploration and exploitation. The results demonstrate that VSCSO performs competitively compared to existing computational and metaheuristic methods, particularly in handling variable interaction strengths. However, there

remains potential for further enhancement. Future work could focus on refining the algorithm's search mechanisms or integrating adaptive strategies to improve performance in more complex or large-scale configurations.

Table 4. Wilcoxon and Friedman Test for Group 1,2 and 3

No	Paired Strategy	Test Statistic: Comparison for VSCSO <	=	>	Total Samples	P-value (Asymp. Sig. (2-tailed))	Friedman Mean Rank Test: Mean Rank	Rank	Null Hypothesis	Conclusion
1.	VSCSO vs VS-TACO	27	17	1	45	0.001	VSCSO – 4.12 VS-TACO – 2.41	5 1	Reject	VS-TACO Outperforms
2.	VSCSO vs TTSGA	25	18	2		0.001	VSCSO – 4.12 TTSGA – 2.74	5 2	Reject	TTSGA Outperforms
3.	VSCSO vs HABC	23	15	7		0.149	VSCSO – 4.12 HABC – 3.32	5 3	Retain	No significant difference
4.	VSCSO vs ABCVS	19	18	8		0.500	VSCSO – 4.12 ABCVS – 3.50	5 4	Retain	No significant difference
5.	VSCSO vs TVG	4	17	24		0.001	VSCSO – 4.12 TVG – 4.90	5 6	Reject	VSCSO Outperforms
6.	VSCSO vs IPOG	27	9	7	43	0.001	VSCSO – 1.27 IPOG – 1.73	1 2	Reject	VSCSO Outperforms
7.	VSCSO vs GVS	13	14	4	31	0.006	VSCSO – 1.65 GVS – 1.35	2 1	Reject	GVS Outperforms
8.	VSCSO vs SCAVS	15	9	6	30	0.026	VSCSO – 1.65 SCAVS – 1.35	2 1	Reject	SCAVS Outperforms
9.	VSCSO vs HABCsm	15	14	1	30	0.006	VSCSO – 1.73 HABCsm – 1.27	2 1	Reject	HABCsm Outperforms
10.	VSCSO vs GAMIPOG	9	12	3	24	0.91	VSCSO – 1.63 GAMIPOG – 1.38	2 1	Retain	No significant difference
11.	VSCSO vs SCIPOG-VS	2	0	21	23	0.01	VSCSO – 1.09 SCIPOG-VS – 1.91	1 2	Reject	VSCSO Outperforms
12.	VSCSO vs DA-FO	5	5	7	17	0.528	VSCSO – 1.97 DA-FO – 1.82	2 1	Retain	No significant difference
13.	VSCSO vs Density	6	6	5		0.320	VSCSO – 1.97 Density – 2.21	2 3	Retain	No significant difference

5. ACKNOWLEDGEMENTS

The author would like to acknowledge the support from the Fundamental Research Grant Scheme (FRGS) under a grant number of FRGS/1/2024/ICT01/UNIMAP/02/1 from the Ministry of Higher Education Malaysia.

6. REFERENCES

[1] N. H. C. Rose, R. R. Othman, H. L. Zakaria, A. J. Suali, and Z. A. Ahmad, "Wingsuit Flying Search Optimization Algorithm Strategy for Combinatorial T-Way Test Suite Generation," *International Journal of Advances in Soft Computing and its Applications*, vol. 16, no. 3, pp. 272–293, 2024, doi: 10.15849/IJASCA.241130.15.

[2] R. Krishnamoorthy, K. Tanaka, and R. Thiagarajan, "Analysis of Obese Patients Using Machine Learning to Categorize Hidden Risk Factors in Explorative Assessment," in *International Exchange and Innovation Conference on Engineering and Sciences*, Kyushu University, 2024, pp. 685–691. doi: 10.5109/7323336.

[3] A. Zakypbekov, W. Xu, T. Niidome, A. Kalbekov, and E. Omurzak, "Effect of some experimental conditions of the pulsed plasma in liquid on the formation of nanostructures," in *International Exchange and Innovation Conference on Engineering and Sciences*, Kyushu University, 2024, pp. 1107–1113. doi: 10.5109/7323396.

[4] N. Anwar and S. Kar, "Review Paper on Various Software Testing Techniques & Strategies," *Global Journal of Computer Science and Technology*, pp. 43–49, May 2019, doi: 10.34257/gjcstcvol19is2pg43.

[5] M. Bajjouk, M. Ehsan Rana, C. Reka Ramachandiran, and S. Chelliah, "Software testing for reliability and quality improvement," 2021.

[6] M. Z. Z. Ahmad, R. R. Othman, M. S. A. R. Ali, and N. Ramli, "A Self-Adapting Ant Colony Optimization Algorithm Using Fuzzy Logic (ACOF) for Combinatorial Test Suite Generation," in *IOP Conference Series: Materials Science and Engineering*, 2020. doi: 10.1088/1757-899X/767/1/012017.

[7] E. Pira and M. Khodizadeh-Nahari, "Combinatorial t-way test suite generation using an improved asexual reproduction optimization algorithm," *Appl Soft Comput*, vol. 150, 2024, doi: 10.1016/j.asoc.2023.111070.

[8] K. M. Htay, H. L. Zakaria, R. R. Othman, N. Ramli, and A. Amir, "A Pairwise T-Way Test Suite Generation Strategy Using Gravitational Search Algorithm," *ICAICST 2021 - 2021 International Conference on Artificial Intelligence and Computer Science Technology*, 2021.

[9] D. R. Kuhn, R. Bryce, F. Duan, L. S. Ghandehari, Y. Lei, and R. N. Kacker, "Combinatorial Testing: Theory and Practice," in *Advances in Computers no. 99*, 2015, ch. 1, p. 99. Accessed: Feb. 23, 2025. [Online]. Available: https://tsapps.nist.gov/publication/get_pdf.cfm?pub_id=918448

[10] Muazu A. A., Hashim A. S., Audi U. I., Aliyu Y., and Yahaya M. S., "Combinatorial Interaction Testing for T-Way Test Case Generation: A Scoping Review of the Perspective Features," *Journal of China University of Mining and Technology*, no. 4, p. 121, 2024, doi: 10.1654/zkdx.2024.29.4-1.

[11] A. Aminu Muazu, A. Sobri Hashim, A. Sarlan, and M. Abdullahi, "SCIPOG: Seeding and constraint support in IPOG strategy for combinatorial t-way testing to generate optimum test cases," *Journal of King Saud University - Computer and Information Sciences*, vol. 35, no. 1, pp. 185–201, Jan. 2023, doi: 10.1016/j.jksuci.2022.11.010.

[12] N. F. Maidin, S. Hassan, S. Baharom, and A. B. Md. Sultan, "A Comparative Study on Testing Optimization Techniques with Combinatorial Interaction Testing for Optimizing Software Product Line Testing," *Journal of Advanced Research in Applied Sciences and Engineering Technology*, no. 1, pp. 77–94, 2025, doi: https://doi.org/10.37934/araset.49.1.7794.

[13] M. Z. Z. Ahmad, R. R. Othman, N. Ramli, and M. S. A. R. Ali, "VS-TACO : A Tuned Version of Ant Colony Optimization for Generating Variable Strength Interaction in T-Way Testing Strategy," in *International Conference on Software and Computer Applications*, 2022. doi: 10.1145/3524304.3524311.

[14] A. Seyyedabbasi and F. Kiani, "Sand Cat swarm optimization: a nature-inspired algorithm to solve global optimization problems," *Eng Comput*, vol. 39, no. 4, pp. 2627–2651, 2022, doi: 10.1007/s00366-022-01604-x.

[15] J. M. Altmemi, R. R. Othman, and R. Ahmad, "Review of combinatorial testing strategy," *International Journal of Advanced Trends in Computer Science and Engineering*, vol. 8, no. 6, pp. 2877–2881, Nov. 2019, doi: 10.30534/ijatcse/2019/31862019.

[16] N. Ramli, R. R. Othman, Z. I. Abdul Khalib, and M. Jusoh, "A Review on Recent T-way Combinatorial Testing Strategy," in *MATEC Web of Conferences*, EDP Sciences, Dec. 2017. doi: 10.1051/matecconf/201714001016.

[17] 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," *J Phys Conf Ser*, vol. 1755, no. 10, 2020.

[18] A. B. Nasser, A. A. Alsewari, N. M. Tairan, and K. Z. Zamli, "Pairwise Test Data Generation Based On Flower Pollination Algorithm," *Malaysian Journal of Computer Science*, pp. 242–257, 2017, Accessed: Feb. 05, 2025. [Online]. Available: https://ejournal.um.edu.my/index.php/MJCS/article/view/7069/4715

[19] J. Arshem, "TVG," SourceForge. [Online]. Available: http://sourceforge.net/projects/tvg

[20] Z. Wang, B. Xu, and C. Nie, "Greedy Heuristic Algorithms to Generate Variable Strength

Combinatorial Test Suite," *Proc Int Conf Qual Softw*, pp. 155–162, 2008, doi: 10.1109/QSIC.2008.52.

[21] Aminu Muzau A., Sobri Hashim A., Danjuma Maiwada U., and Muppidi A., "Enhanced Version of Seeding and Constraint support in IPOG strategy for Variable Strength Interaction T-way Testing," *Malaysian Journal of Computer Science*, vol. 36, no. 4, pp. 44–53, 2023, doi: 10.22452/mjcs.vol36no4.3.

[22] Y. Lei, R. Kacker, D. R. Kuhn, V. Okun, and J. Lawrence, "IPOG: A General Strategy for T-Way Software Testing," in *Proceedings of the International Symposium and Workshop on Engineering of Computer Based Systems*, 2007, pp. 549–556.

[23] R. R. Othman, K. Zuhairi Zamli, and L. E. Nugroho, "General variable strength t-way strategy supporting flexible interactions," *Maejo Int. J. Sci. Technol*, vol. 6, no. 03, pp. 415–429, 2012, [Online]. Available: www.mijst.mju.ac.th

[24] A. Sharma, S. Sharma, and D. Gupta, "Ant Colony Optimization Based Routing Strategies for Internet of Things," *EVERGREEN Joint Journal of Novel Carbon Resource Sciences & Green Asia Strategy, Vol. 10, Issue 02, pp998-1009, Kyushu University Institutional Repository*, vol. 10, no. 2, pp. 998–1009, Jun. 2023, doi: 10.5109/6793654.

[25] P. A. Kowalski, S. Kucharczyk, and J. Mańdziuk, "Constrained Hybrid Metaheuristic Algorithm for Probabilistic Neural Networks Learning," *ArXiv*, Jan. 2025, [Online]. Available: http://arxiv.org/abs/2501.15661

[26] A. K. Alazzawi *et al.*, "HABCSm: A Hamming Based t -way Strategy based on Hybrid Artificial Bee Colony for Variable Strength Test Sets Generation," *International Journal of Computers, Communications and Control*, vol. 16, no. 5, pp. 1–18, 2021, doi: 10.15837/ijccc.2021.5.4308.

[27] A. K. Alazzawi, H. M. Rais, and S. Basri, "HABC: Hybrid artificial bee colony for generating variable T-way test sets," *Journal of Engineering Science and Technology*, vol. 15, no. 2, pp. 746–767, 2020.

[28] M. I. Younis, "GAMIPOG: A DETERMINISTIC GENETIC MULTI-PARAMETER-ORDER STRATEGY FOR THE GENERATION OF VARIABLE STRENGTH COVERING ARRAYS," *Journal of Engineering Science and Technology*, vol. 15, no. 5, pp. 3142–3161, 2020.

[29] J. M. Altmemi, R. R. Othman, and R. Ahmad, "SCAVS: Implement Sine Cosine Algorithm for generating Variable t-way test suite," in *IOP Conference Series: Materials Science and Engineering*, IOP Publishing Ltd, Sep. 2020. doi: 10.1088/1757-899X/917/1/012011.

[30] A. K. Alazzawi, H. M. Rais, and S. Basri, "ABCVS: An Artificial Bee Colony for Generating Variable T-Way Test Sets," *International Journal of Advanced Computer Science and Applications*, vol. 10, no. 4, pp. 259–274, 2019, doi: 10.14569/ijacsa.2019.0100431.

[31] D. Wu, H. Rao, C. Wen, H. Jia, Q. Liu, and L. Abualigah, "Modified Sand Cat Swarm Optimization Algorithm for Solving Constrained Engineering Optimization Problems," *Mathematics*, vol. 10, no. 22, Nov. 2022, doi: 10.3390/math10224350.

[32] M. Z. Z. Ahmad, "A Self-Adapting Ant Colony Optimization Algorithm using Fuzzy Logic (ACOF) for Combinatorial Test Suite Generation," Universiti Malaysia Perlis, 2023.