t-way test data generation startegy with MBO algorithm

Hasneeza L. Zakaria
School of Computer and Communication Engineering
Universiti Malaysia Perlis(UniMAP)
02600 Arau, Perlis, MALAYSIA
hasneeza@unimap.edu.my

Kamal Z. Zamli
Faculty of Science Computer and Software Engineering
Universiti Malaysia Pahang(UMP)
Lebuhraya Tun Razak, 26300 Kuantan, Pahang, MALAYSIA
Kz.zamli@ump.edu.my

***Abstract*—This paper presented the implementation of a nature inspired metaheuristic search algorithms that are Migrating Birds Optimization (MBO) algorithm and Genetic Algorithm (GA) hybrid to a t-way test data generation strategy. The proposed strategy is called improved MBO Testing Strategy (iMTS). Based on the published benchmarking results, the result of these strategies is competitive with most existing strategies in terms of the generated test size in many of the parameter configurations. For a higher strength, iMTS is able to produce a minimum test suite size. In the case where these strategies are not the most optimal, the resulting test size is sufficiently competitive. The strategy serves as our research conduit to investigate the effectiveness of MBO algorithm for t-way test data generation strategy.**

Keywords-t-way testing, MBO Algorithm, GA, iMTS

I. INTRODUCTION

Modern software systems are usually complex and have large number of configurations. It is impossible to test exhaustively as the number of configurations grows exponentially with the number of configuration options. Thus, there are no sufficient resources and time to test each possible combination of option settings for every combination. Interaction testing approaches such as t-way testing strategies use sampling method to test selected configurations where each possible combination of option settings for every combination of *t* options appears at least once[1].

Existing t-way strategies adopt many different approaches, such as pure computational-based approaches like Jenny[2], TConfig[3] and IPOG[4] and also AI-based approach that mostly used nature inspired metaheuristics algorithm. Nature inspire metaheuristic algorithm have been popular in solving myriad optimization problems in multiple fields such as engineering, networking, data mining and industrial[5].

In the past 15 years, researchers in t-way testing also have been using nature inspired algorithms in finding the minimum set of test cases. The first 10 years were focused on pairwise and 3-way data generation strategies to test on small size data; mostly with $t \leq 3$ by implementing trajectory based algorithms such as Simulated Annealing(SA), Tabu Search(TS) and Hill Climbing(HC) and classics population based algorithm such as Genetic Algorithm (GA) and Ant Colony Algorithm (ACA)[6]. In the previous 5 years, researchers have been innovatively trying to explore the higher strength $(t > 6)$ data generation strategies. This were possible with the creation of new nature inspired algorithms that are mostly population based such as Artificial Bee Colony(ABC), Bee Algorithm(BA) and Bat Algorithm[7]. Population based algorithms has a global exploration and local exploitation mechanism[8]. Hence they yields a better result compared to trajectory based algorithms. This can be further enhanced by using hybrid algorithm where two or more algorithm are integrate and the execution control is based on rule based algorithm[5].

This paper discusses the design, implementation and evaluation of MBO algorithm and GA hybrid based strategy i.e. iMTS. Based on the published benchmarking results, the iMTS performs competitively with existing strategies(SA, GA, ACA[6], BA[9], Jenny[2], TConfig[3], IPOG[4], PPSTG[10] and PHSS[11]) and managed to get the smallest test suite the same size as PHSS when $t > 4$.

This paper is organized as follows. In section 2, the MBO algorithm and GA are given. In section 3, the iMTS is presented. In section 4, the experimental results and comparison of results of the proposed algorithm with other algorithms are also presented. Finally, the last section concluded our work.

II. THE MBO ALGORITHM AND GA

A. Migrating Birds Optimization(MBO) Algorithm

MBO is inspired from the long distance flight of gregarious birds such as ibises, pink-footed geese and Canadian geese that usually fly in a V formation during winter migration proposed by Duman et al.[12]. It is found that in the V formation, energy savings can be achieved by using the aerodynamic up wash produced by the preceding bird[13]. Theoretically, birds could save more than 50% of their energy by flying in V formation compared to flying solo.

MBO algorithm is a neighborhood search based algorithm where solutions were improved from exploring the neighborhood based on the benefit sharing mechanism of the

V formation. There are 4 phases in MBO i.e.(1) Initialization, (2) improve the leader, (3) improve the follower and (4) select a new leader. It also has five important parameters:

n = number of initial solutions (flock size),
k = number of neighbor solutions to be considered (the speed of flight),
x = number of neighbor solutions to be shared with the next solution (wing-tip span),
m = number of tours (the number of wing flaps),
K = number of iteration

The first phase is to initialize an n size flock randomly and choose a leader bird from the flock. The rest of the solutions *(n-1)* are assigned to the right and left list arbitrarily to form V formation. The second phase is to improve the leader solution with k neighbor solutions. If one of the neighbor solutions is better than the current leader solution, the neighbor solution will replace the leader solution. The unused (neighbors that not been used for improvement), *k-1* best unused neighbor solutions is shared with the next follower solutions on the left and right.

The third phase is to improve the follower solution on the left with *k-x* solutions. This solution is combined with the unused *k-1* solutions from leader solutions. The best solution replaced the current solution if it is better than the current solution. The remaining *k-1* solutions will be shared with the next follower solution on the left. This procedure will be repeated for the solution on the right.

The fourth and last phase is to select a new leader solution. The leader solution is moved alternately to the ends of the left and the right list, and the first solution in the corresponding list is forwarded as the new leader. It is like in a V formation; the strongest bird leads the flock and is relocated to the end of the line when tired while the immediate next bird will take the lead.

There are a few researchers who has modified the MBO algorithm to adapt to their problems and to improve its performance i.e. a modified MBO for credit card fraud detection system at Turkish bank[14], an enhanced MBO(EMBO) to solve no-wait flowshop sequencing problem[15], a new cooperative and modified variant of MBO[16], an improved MBO(IMBO) to solve hybrid flowshop scheduling problem[17], MBO to solve maritime container problems[18], a modified MBO(M-MBO) to solve university course timetabling problem[19] and a modified MBO(MMBO) to solve close loop layout problem[20]. Their results show that the modified MBO performed better than the original MBO algorithm.

MBO algorithm feature a number of solutions running in parallel and the benefit sharing mechanism between the solutions. Benefit sharing mechanism is where the best unused neighbor from the previous solution was shared with the next solution. These features could be harnessed to solve combinatorial testing problems as they were proven to be effective in solving various combinatorial optimization problems.

B. Genetic Algorithm(GA)

The GA was introduced by Holland and mimics Darwinian Theory of the survival of the fittest.

GA solves optimization problems by manipulating initial population (individual chromosomes sampled randomly). Each chromosome is evaluated based on a fitness function which is related to its success in solving a given problem. Given an initial population of chromosomes, GA proceeds by choosing chromosomes to serve as parents and then replacing members of the current population with new chromosomes that are copies of the parents i.e. offspring. The process of selection and population replacement goes on until a stopping criterion (achieving effective test data) has been met[21]. In our case, a test case is the same as a chromosome. This process can be sum up as the GA phases of (1) selection, (2) crossover and (3) mutation.

Many t-way testing strategies adopted GA[22],[23],[24],[25] and [26]. These strategies address the low strength $(t \leq 3)$. Hence, we cater for $t \geq 6$ by using our MBO-GA hybrid in iMTS strategy.

III. iMTS STRATEGY

In this section, our t-way strategy with improved MBO algorithm which is a MBO-GA hybrid is introduced i.e. iMTS. The iMTS algorithm is as illustrated in Fig. 1.

The iMTS generates the interactions elements list containing all interactions tuple combinations for each pair for a P-valued parameter after accepting the input parameters and their corresponding value. While the interaction elements list is not empty, a current test case is generated. The improved MBO algorithm generates the neighbor test case (the leader bird that carries the largest weight). The weight of the current test case and neighbor test case are calculated. Test case with a larger weight will be the best test case. The pair pertaining to the best test case will be removed from the interaction elements list and the best test case will be stored in the test suite.

The improvements to the original MBO are depicted as follows:

A. Multiple neighborhood structure

IMTS uses four neighborhood structures; which are the random search neighborhood structure (1) that traverses the neighborhood at random, the partition based neighborhood structure (2) that divide the neighborhood into 4 parts where each part is accessed sequentially, the maximum swap neighborhood structure (3) that swaps the random search neighborhood structure and partition based neighborhood structure and returns the maximum solution and the random walk neighborhood structure (4) which is inspired by the one-dimensional random walk. In this neighborhood, a walker moves in counterclockwise manner and at each step moves +1 or -1 with equal probability of a quarter.

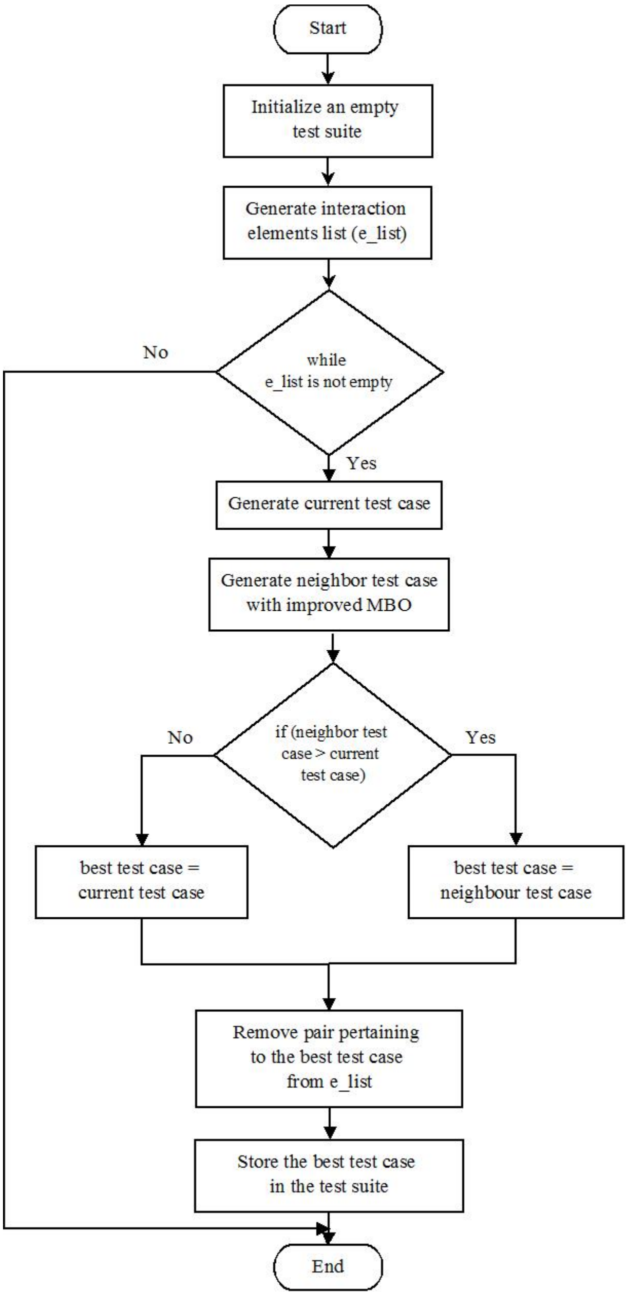

Fig. 1. iMTS algorithm flowchart

B. *Initialization*

The original MBO[12] randomly initialize the initial flock. For diversity, instead of generating the initial flock at random we generated the initial flock with the random walk neighborhood structure.

C. *Elitism*

Elitism were introduced by De Jong as a simple strategy to ensure the survival of the best solution by preserving it in the next flock[27]. Elitism is used to store thirty three percent of the best solutions from the previous run and inserted them back into the next flock. This ensures that the next flock has better solutions to choose from. Two random test cases were chosen from the population and the one with more weight is chosen as the best test case. This best test case will replace the poorest test case in the previous population. Then, the best test cases from the previous population were added to the elitist array.

D. *Genetic Algorithm(GA)*

GA is used to enhance the performance of iMTS by performing selection, crossover, mutation and evaluation on the best solution. GA enhanced the quality of the flock and also discovers a better solution space.

E. *Iterated Local Search(ILS)*

ILS is implemented to increase the convergence speed and to escape from local optima[19]. There are two phases in the ILS i.e. getting the best solution and comparing the best solution with the neighbor solution. A best solution is generated after the leader replacement and after the sorting of the follower birds from the flock. Another best solution is also generated from a neighbor flock and compared with the solution from the current flock. If the neighbor flock solution is better, then it will replace the best solution from the flock.

Fig. 2. illustrated the improvements made to the MBO algorithm.

IV. RESULT AND DISCUSSION

Parameter tuning for the improved MBO algorithm in iMTS took the same important parameters as Duman et al.[12] into consideration i.e. number of birds; *n*, number of neighbor solutions; *k*, number of shared neighbor solutions; *x*, number of tours; *m* and number of iteration; *K* as in MTS. However, with the improvements of crossover, mutation, elitism and ILS, the most minimum value can be set for *n, k, x, m* and *K*. This tuning process is crucial in finding the best parameters value that can yield the smallest test suite size.

A series of experiments are conducted to find the best parameters for the improved MBO in iMTS with system configuration consisting of 5 10-valued parameters configuration where $t = 2$. The iMTS was run 20 times with the same value of *n, k, m, x* and *K* i.e. $n=7$, $k = 3$, $m = 1$, $x = 2$ and $K = 1$ against different elitism range of 33%, 50% and 80%. The number of ILS loop was also varied from 330, 800, 1000, 1330 and 1500. The results of the experiments are shown in Fig. 3. and Fig. 4. respectively.

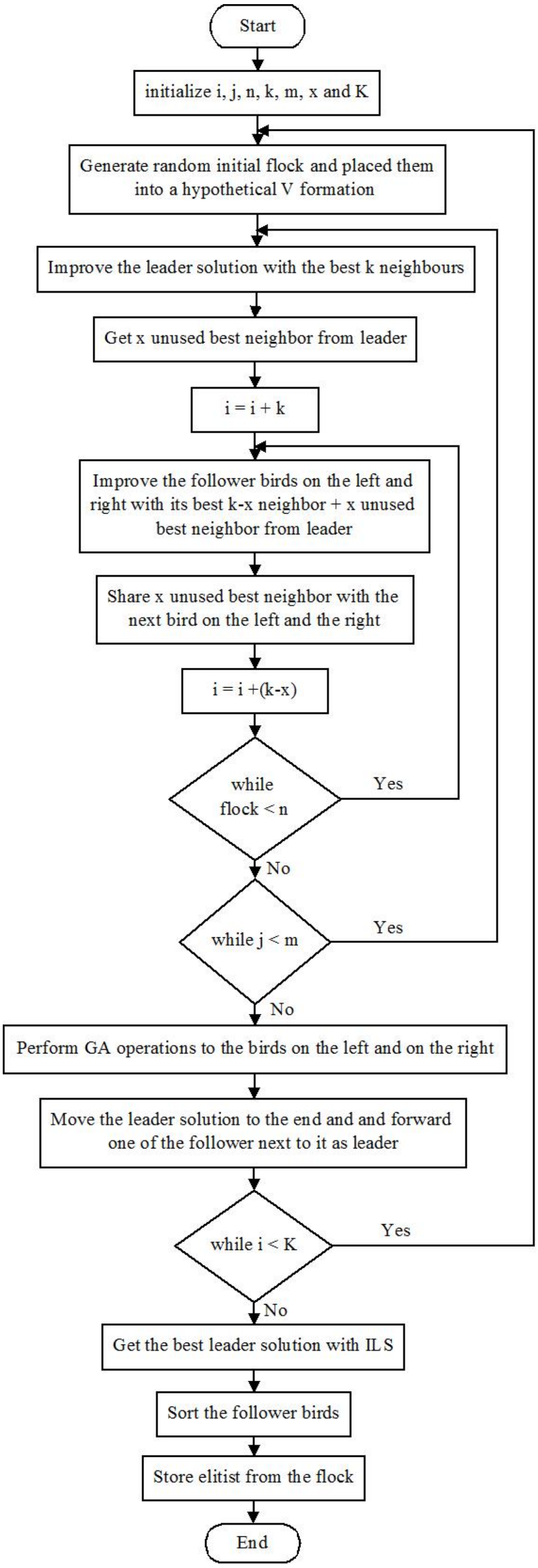

Fig. 2. Improved MBO algorithm

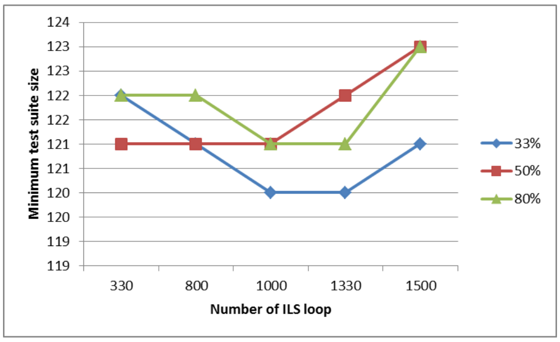

Fig. 3. Minimum test suite size vs. number of ILS loop

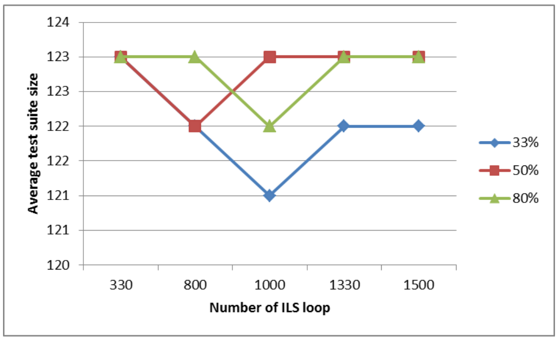

Fig. 4. Average test suite size vs. number of ILS loop

The occurrence of generated test suite size equals to 120 is 25% when the ILS loop is 1000 and elitism = 33% as shown in Fig. 3. Hence it can be considered as the smallest test suite size generated. The smallest test suite average size is 121, also when ILS loop is 1000 and elitism = 33% as shown in Figure 4. Thus the ideal parameters for iMTS are shown in Table I. because these parameters generate the minimum and average test suite size.

iMTS were benchmarked against SA, GA, ACA[6], BA[9], Jenny[2], TConfig[3], IPOG[4], PPSTG[10] and PHSS[11] as adopted from Hazli and Zamli[9].

To aide in the result discussion, covering arrays (CA) notation is explained. Interaction test suites can be represented with CA notation. The CA has four parameters; N, t, P, and v (i.e., CA (N, t,v^P)). Here, the symbols P, v, and t are used to refer to number of parameters, values, and interaction strength for the CA, respectively. For example, CA (N, 2, 2^{10}) represents a test suite that covers 2-way interaction for a system with ten 2-value parameters.

Table II. result's reveals that iMTS generate a good result comparable to all AI-based strategies for t = 2 and t = 3. iMTS managed to generate the smallest test suite size for CA(N,2,13,3) and CA(N,2,10,5) where the size is 16 and 43 respectively. For CA(N,2,13,3) it is the same result as generated by SA and for CA(N,2,10,5) it is the same result as PHSS. Whereas, for CA(3,6,5), the test suite size generated is 198, the same size as generated by BA. The test suite size for CA(N,3,7,5) is 222 which is not so far from the test suite size generated by GA and ACA, which is 218.

TABLE I. PARAMETER SETTINGS FOR IMTS

Parameter	iMTS
Number of initial solutions, n	7
Number of neighbor solutions to be considered, k	3
Number of neighbor solutions to be shared with the next solution, x	2
Number of tours, m	2
Number of iteration, K	1
Number of ILS loop	1000
Elitism percentage	33

TABLE II. IMTS VS. OTHER NATURE INSPIRED STRATEGIES

CA	SA	GA	ACA	PSTG	PHSS	BA	iMTS
CA(N,2,3^4)	9	9	9	9	9	9	9
CA(N,2,3^{13})	16*	17	17	17	18	19	16*
CA(N,2,10^{10})	NA	157	159	NA	155	183	174
CA(N,2,5^{10})	NA	NA	NA	45	43*	47	43*
CA(N,3,3^6)	33	33	33	42	39	42	41
CA(N,3,4^6)	64	64	64	102	70	108	104
CA(N,3,5^6)	152	125	125	NA	199	198	198
CA(N,3,5^7)	201	218	218	229	236	227	222

TABLE III. CA(N,T,2^{10}) WITH T VARIED FROM 2 TO 6

CA	IPOG	Jenny	TConfig	PSTG	PHSS	BA	iMTS
CA(N,2,2^{10})	10	10	9	8	7	8	8
CA(N,3,2^{10})	19	18	20	17	16*	18	16*
CA(N,4,2^{10})	49	39	45	37	37	39	38
CA(N,5,2^{10})	128	87	95	82	81	85	80*
CA(N,6,2^{10})	352	169	183	158	158	162	159

Table III. result's shows the comparison of ten 2 parameters value with strength ranging from 2 to 6 i.e. CA(N,t,10,2). It is depicted that iMTS was able to compete with the benchmarked strategies especially with PSTG[10] and PHSS[11]. For CA(N,3,10,2), iMTS yielded the smallest test suite size i.e. 16 the same as generated by PHSS. For CA(N,5,10,2) the test suite size is 80 which is the smallest test suite size generated compared to the other strategies.

The weakness of MBO is early convergence i.e. early termination could take place before the feasible region is thoroughly explored and thus the result obtained is not optimal when the search space increases in size[14]. It is apparent from these findings that by incorporating GA with MBO, this weakness can be overcome.

V. CONCLUSION

Experimental results irrefutably show that the improvement of iMTS has enabled it to improve MBO algorithm's global exploration and local exploitation. The hybridization of MBO algorithm with GA has widened the search space and enables the algorithm to search thoroughly and avoid it from getting trapped in local optima. The introduction of elitists from previous population to the next population has retained the best test cases from the previous population.

ACKNOWLEDGMENT

The preferred spelling of the word “acknowledgment” in America is without an “e” after the “g.” Avoid the stilted expression “one of us (R. B. G.) thanks ...”. Instead, try “R. B. G. thanks...”. Put sponsor acknowledgments in the unnumbered footnote on the first page.

REFERENCES

[1] C. Yilmaz, E. Dumlu, M. B. Cohen, A. Porter, and S. Member, "Reducing Masking Effects in Combinatorial Interaction Testing : A Feedback Driven Adaptive Approach," *IEEE Transactions on Software Engineering,* vol. 40, pp. 43-66, 2014.

[2] B.Jenkins. *Jenny Test Tools*. Available: http://www.burtleburtle.net/bob/math/jenny.html

[3] A. Williams. *TConfig*. Available: http://www.site.uottawa.ca/~awilliam/

[4] 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.

[5] K. Z. Zamli, "Software Testing Inspiration From Nature," ed. Kuantan, Pahang: Penerbit Universiti Malaysia Pahang, 2015, pp. 1-62.

[6] W. Afzal, R. Torkar, and R. Feldt, "A systematic review of search-based testing for non-functional system properties," *Information and Software Technology,* vol. 51, pp. 957-976, 2009.

[7] B. Xing and W.-J. Gao, *Intelligent Systems Reference Library 62 Innovative Computational Intelligence : A Rough Guide to 134 Clever Algorithms*. Switzerland: Springer International Publishing, 2014.

[8] Z. Beheshti, S. Mariyam, and H. Shamsuddin, "A Review of Population-based Meta-Heuristic Algorithm," *International Journal of Advances in Soft Computing and its Applications,* vol. 5, pp. 1-35, 2013.

[9] M. Hazli, M. Zabil, and K. Z. Zamli, "Implementing a T -Way Test Generation Strategy Using Bees Algorithm," *International Journal of Advances in Soft Computing and its Applications,* vol. 5, pp. 119-126, 2013.

[10] B. S. Ahmed, K. Z. Zamli, and C. P. Lim, "Constructing a T-Way Interaction Test Suite Using The Particle Swarm Optimization Approach," *International Journal of Innovative Computing, Information and Control,* vol. 8, pp. 431-451, 2012.

[11] 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, pp. 553-568, 2012.

[12] E. Duman, M. Uysal, and A. F. Alkaya, "Migrating Birds Optimization: A new metaheuristic approach and its performance on quadratic assignment problem," *Information Sciences,* vol. 217, pp. 65-77, 2012.

[13] B. Voelkl, M. Unsöld, J. R. Usherwood, and A. M. Wilson, "Matching times of leading and following suggest cooperation through direct reciprocity during V-formation flight in ibis," in *Proceedings of National Academy of Sciences of the United States of America*, 2015, pp. 2115-2120.

[14] E. Duman, A. Buyukkaya, and I. Elikucuk, "A Novel and Successful Credit Card Fraud Detection System Implemented in a Turkish Bank," *2013 IEEE 13th International Conference on Data Mining Workshops,* pp. 162-171, 2013.

[15] K. Z. Gao and P. N. Suganthan, "An Enhanced Migrating Birds Optimization Algorithm for No-wait Flow Shop Scheduling Problem," in *IEEE Symposium on Computational Intelligence in Scheduling (CISched)*, ed, 2013, pp. 9-13.

[16] H. Makas, N. Yumusak, and A. T. A. B. C. Algorithm, "New Cooperative and Modified Variants of the Migrating Birds Optimization Algorithm," in *International Conference on Electronics, Computer and Computation*, Ankara, Turkey, 2013, pp. 176-179.

[17] Q.-k. Pan and Y. Dong, "An improved migrating birds optimisation for a hybrid flowshop scheduling with total flowtime minimisation," *Information Sciences,* vol. 277, pp. 643-655, 2014.

[18] E. Lalla-ruiz, C. Exp, and J. D. Armas, "Migrating Birds Optimization for the Seaside Problems at Maritime Container Terminals," *Journal of Applied Mathematics,* pp. 1-21, 2014.

[19] L. W. Shen, H. Asmuni, and F. C. Weng, "Migrating Birds Optimization in Solving University Timetabling Problem," *Jurnal Teknologi,* vol. 1, pp. 89-96, 2015.

[20] S. Niroomand, A. Hadi-Vencheh, R. Şahin, and B. Vizvari, "Modified migrating birds optimization algorithm for closed loop layout with exact distances in flexible manufacturing systems," *Expert Systems with Applications,* 2015.

[21] Y. Suresh and S. Rath, "A Genetic Algorithm based Approach for Test Data Generation in Basis Path Testing," *The International Journal of Soft Computing and Software Engineering(JSCSE),* vol. 3, pp. 326-332, 2014.

[22] J. Stardom, "Metaheuristics and the search for covering and packing array," M.S. Thesis, Department of Mathematics, Simon Fraser University, 2001.

[23] M. B. Cohen, P. B. Gibbons, W. B. Mugridge, and C. J. Colbourn, "Constructing test suites for interaction testing," in *25th International Conference on Software Engineering, 2003. Proceedings.*, 2003, pp. 38-48.

[24] T. Shiba, T. Tsuchiya, and T. Kikuno, "Using artificial life techniques to generate test cases for combinatorial testing," in *Proceedings of the 28th Annual International Computer Software and Applications Conference, 2004. COMPSAC 2004.*, 2004, pp. 1-14.

[25] J. D. McCaffrey, "Generation of Pairwise Test Sets Using a Genetic Algorithm," presented at the 2009 33rd Annual IEEE International Computer Software and Applications Conference, 2009.

[26] P. Flores and Y. Cheon, "PWiseGen : Generating Test Cases for Pairwise Testing Using Genetic Algorithms PWiseGen : Generating Test Cases for Pairwise Testing Using Genetic Algorithms," in *2011 IEEE International Conference on Computer Science and Automation Engineering(CSAE 2011)*, Shanghai, China, 2011.

[27] M. Gendreau and J.-Y. Potvin, *Handbook of Metaheuristics* vol. 146. New York: Springer International Publishing, 2010.