Parameters Tuning of Hybrid Artificial Bee Colony Search based Strategy for *t*-way Testing

Ammar K Alazzawi, Helmi Md Rais, Shuib Basri

Abstract: Hybrid Artificial Bee Colony (HABC) Strategy is latterly developed based on hybridize of an artificial bee colony (ABC) algorithm with a particle swarm optimization (PSO) algorithm. In order to ensure that HABC could perform for t-way testing as useful as other strategies to generate best performance, there are a number of parameters of an HABC algorithm such as the number of colony size (NBees), the number of food source, limit, the number of cycles (maxCycle), weight factor (w) and learning factors (C1, C2) that required to be tuned. In this paper, the process of parameters tuning for hybrid artificial bee colony algorithm has been shown as well as t-way testing, where has been adopted a standard covering array CA (N, 2, 57). The obtained experiment results illustrate that HABC strategy can generate the most minimum and sufficiently results compared to other strategies.

I. INTRODUCTION

Searching the best solution in a large search space among all feasible solutions is an optimization problem. Optimization problem has been used widely in different disciplines such as mechanical and software engineering, etc. Meta-heuristics strategies have outperformed in this regards. In the last two decades, various helpful strategies have been developed based on meta-heuristic algorithms in the related literature such as Particle Swarm Optimization [1, 2],Ant Colony Algorithm (ACA)[3], Harmony Search [4], Late Acceptance Hill Climbing [5], Artificial bee colony algorithm [6-8], Bat-Testing Strategy [9-15], etc. These algorithms considered as part of random algorithms that scout the search space by trial and error effectively. Meta-heuristic algorithms unlike traditional algorithms, where these algorithms considered as population search-based algorithms. Finding a best solution is started from searching the solution space. Then, the candidate best solution is one of the population members.

Revised Manuscript Received on March 08, 2019.

Ammar K Alazzawi, Department of Computer and Information Sciences, Universiti Teknologi PETRONAS, Bandar Seri Iskandar 32610, Perak, Malaysia. Software Quality and Quality Engineering (SQ^2E) Research Cluster, UniversitiTeknologi PETRONAS, 32610 Seri Iskandar, Perak, Malaysia, ammar_16000020@utp.edu.my

Helmi Md Rais, Department of Computer and Information Sciences, Universiti Teknologi PETRONAS, Bandar Seri Iskandar 32610, Perak, Malaysia, helmim@utp.edu.my

Shuib Basri, Department of Computer and Information Sciences, Universiti Teknologi PETRONAS, Bandar Seri Iskandar 32610, Perak, Malaysia. Software Quality and Quality Engineering (SQ^2E) Research Cluster, Universiti Teknologi PETRONAS, 32610 Seri Iskandar, Perak, Malaysia, shuib_basri@utp.edu.my

Regarding *t*-way strategies, the recent works of *t*-way testing have been adopted the meta-heuristics algorithms as the key algorithm for generating the optimal solution. Many existing-based strategies in the literature have been developed based on meta-heuristic algorithms such as Simulated Annealing [16], Particle Swarm Optimization [17], Genetic algorithm [16], Artificial Bee Colony algorithm [18] and so on.

The aforementioned algorithms above has a number of parameters. Therefore, the performance of these algorithms is totally dependent on their parameters tuning. For example, Cuckoo search (CS) demands to tune the parameters like probability factor (*pa*), Nest size (*nest_size*) and Repetition (*N*). Artificial Bee Colony algorithm demands to tune the number of colony size (*NBees*), the number of food source, limit and the number of cycles (*maxCycle*). Particle Swarm Optimization demands to tune the weight value (*w*) and learning factors (*C*1 and *C*2) and population size. As well as for Genetic algorithm, Harmony Search, Simulated Annealing and so on as shown in table 1. The most major challenge of a meta-heuristics algorithm is based on choose the parameter values by the user. As a result, achieving a good performance is dependent on the selected parameter values.

Recently proposed by Karaboge in 2005, one of the meta-heuristics algorithms called Artificial Bee Colony (ABC) algorithm. [18]. Continuation to our previous work [6-8], hybridize Artificial Bee Colony (ABC) algorithm with Particle Swarm Optimization (PSO) is proposed. Hybrid Artificial Bee Colony (HABC) algorithm needs to tune the parameters. In HABC, choosing the value of parameters an important factor of influence on the algorithm performance because of these values, the best solution will be generated as well as implementation time by explores the search space solution globally and exploits the regions locally.

Continuing to the existing work, HABC algorithm is adopted in our research work for *t*-way test suite generation. In this paper, the tuning of HABC will illustrate for *t*-way testing. The remainder of this paper is structured as follows: an overview of HABC algorithm in section 2. Define combinatorial interaction testing in section 3. In section 4 & 5, HABC strategy parameters tuning and experimental results are conducted respectively. Finally, conclusion and future work illustrate in section 6.

II. AN OVERVIEW OF ARTIFICIAL BEE COLONY ALGORITHM

Recently proposed by Karaboge in 2005, one of the meta-heuristics algorithms called Artificial Bee Colony (ABC) algorithm based on foraging behavior of a honeybee colony[18]. The ABC performs some specific tasks by three kinds of bee. These bees are specialized to increase the nectar amount inside the hive by utilized division operation to the population bee and organize them. The three kinds of bees are employed bees, onlooker bees and scout bees, where both onlookers and the scout's bee called unemployed bees. Employed bees represent the half colony

bee, either the other half of the colony represented by onlooker bee. The responsibilities of employed bees are to exploit the food source that explored in advance with higher nectar, and communicate with onlooker bee that waiting at the hive in order to get the information about the food source such as direction, distance and profitability.

Table. 1 Algorithm Parameters for strategies

Algorithm	Parameters
PSO	Max Iteration
	Population Size
	Learning Factors (*C*1and *C*2)
	Weight Factor (*w*)
ABC	Max Iteration
	The number of colony size (*NBees*)
	The number of food source
	Limit
	The number of cycles (*maxCycle*)
CS	Max Iteration
	Population Size (*nest_size*)
	Probability (*pa*)
GA	Max iteration
	Best cloned
	Population size
	Random crossover
	Tournament selection
	Max stale period
	Mutation rate
	Escape mutation
SA	Max iteration
	Cooling schedule
	Starting temperature
ACA	Iteration
	Number of ants
	Pheromone persistence
	Pheromone control
	Heuristic control
	Pheromone amount
	Initial pheromone
	Max stale period
	Elite ants
HS	Improvisation
	Pitch adjustment rate
	Harmony memory consideration rate
	Harmony memory size

Therefore, the onlooker bee select the food source based on the shared information that given by the employed bees. Either scout bee, are a few employed bees that became a scout bee to search the environment randomly in order to detect a new or better than the existing food source. The intelligent behavior of artificial bee colony can be illustrated as follows:

1. Initial phase: - the algorithm starts randomly search the environment to get the food source, within the range of the boundaries of the algorithm's parameters; producing the initial food sources is defined by using Eq (1).

$$x_{ij} = x_{min,j} + \text{rand}(0,1)(x_{max,j} - x_{min,j}) \quad (1)$$

2. Employed bee phase: - in accordance with the detected food sources, the employed bee begins to tap the detected food source. Therefore, the number of employed bee is equal to the number of food source (where each employed bee connected to one food source only). The employed bee collects the nectar and comes back to the hive to conveyance the nectar, and back again to the same source or shares the information of the source with other bees waiting inside the hive by dancing way in the dance area. Then, the employed become a scout's bee after exhausted the nectar of source, and start again to search randomly for a better or new source. Search for a new food source is defined by using Eq (2)

$$V\,ij = X\,i,j + rand\,[-1, 1]\,(X\,ij - X\,kj) \qquad (2)$$

After detecting the food source, the probability of selecting food source is defined by using Eq (3).

$$fitness_i = \begin{cases} \frac{1}{1+f_i}, & if\ f_{i\geq 0} \\ 1 + |f_i|, & if\ f_{i<0} \end{cases} \qquad (3)$$

3. Onlooker bee phase: - the onlooker bee criteria of the food source's selection is based on the nectar amounts; the evaluation of the nectar amounts by given information on the dance area by employed bee. The probability selection of the food source is defined by using Eq (4)

$$Pi = \frac{fit_i}{\sum_{n=1}^{sn} fit_n} \qquad (4)$$

4. Scout bee and Limit phase: - after fulfilled both employed and onlooker bees their tasks, the algorithm inspects the environment in case there is exhausted source to be deserted. The taken decision to abandoned the food source is based on counters called Limit is defined by using Eq (5). During the search, the counter value will updated by the algorithm, if the counter value higher the limit (known as control parameter), then the associated food source with value of the counter will suppose abandoned. The discovered new source by scout bee, will replace with the abandoned food source. All the interactions of the bees has represented as a flowchart on Fig. 1.

$$limit = c.ne\ .D \qquad (5)$$

III. HYBRID ARTIFICIAL BEE COLONY ALGORITHM

In the last two decade, swarm intelligence (SI) optimization algorithms are the most effective and interesting algorithms, where the inspiration of swarm intelligence concept came through observing the birds' flocks, fish schools, and bee colonies and insect colonies such as termites, ant when they are looking for food. Thus, the focus of recent works is mostly of the use of these algorithms to solve a real-world optimization problem such as the Artificial Bee Colony (ABC) algorithm [18], Bacterial Foraging Optimization (BFO) [19], Particle Swarm Optimization (PSO) [17], etc. One of these algorithms is bee's colony optimization algorithm that has been used to solve optimization problems, where has several characteristics that can be carried out so that the intelligent search systems can be modelled like the "queen bee, task selection, bee foraging, navigation systems, nest site selection, mating, collective decision making, bee dance (communication), floral/pheromone laying" [20]. ABC algorithm exactly like any popular heuristic algorithm "as a result of the randomization it includes in the first steps" algorithm; ABC is like other algorithms has advantages and disadvantages.

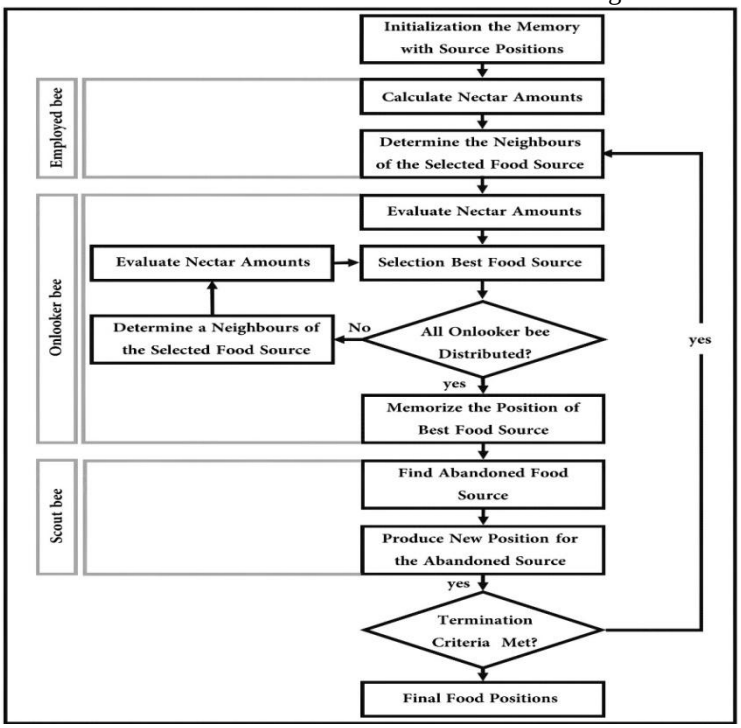

Fig. 1 Flowchart of the Artificial Bee Colony algorithm

One of the most ABC algorithm weaknesses is the ABC simple operation that completely depends on the solution development process there are insufficiencies; the speed of convergence of the algorithm is increased [20]. Thus, a fast convergence will be occurred in the algorithm, and this can be led to be stuck for some of the complex problems in the local optimum. On the other hand, the procedure of information sharing activity is being done on a one dimension having a random neighbour in each solution improvement; therefore, will be decreased the speed of convergence as increased of the problem's dimension [21]. In addition, the information sharing activity of ABC algorithm showed a weak performance in the experiments [22]. However, it is impossible can be find an optimization algorithm, which can get the global optimum for all the optimization problem. To overcome these disadvantages, several ABC algorithms' variants have to promote the convergence speed, exploitation and exploration capability, and escape being confined at the local optimum [20, 21]. The modifications involved hybridizing with other optimization algorithms or changing on ABC itself. Several of the modifications focus on the information sharing process or improvement of the ABC's solution, which is the major factor involved in the exploitation and exploration capability of the algorithm.

In this paper, our inspiration to improve ABC algorithm comes from movement operation of particles in the PSO algorithm. The processes of information sharing and solution improvement mechanism of PSO are unique and differ from the exist one in the ABC, where consists of the unique and special parameter called Weight Factor (*w*). The function of a velocity parameter is to control the required improvement degree based on the previous solution. In addition to velocity, there are two other parameters called learning factors (*C*1 and *C*2), where there factors function are to determine the relative influence of cognitive (self-confidence) and social (swarm-confidence) components, respectively (Eq. (6)).

$$V_{i,d}^{t+1} = W^t * V_{i,d}^t + C_1^t * r_1 * (pbest_{i,d}^t - X_{i,d}^t) + C_2^t * r_2 * (gbest_{i,d}^t - X_{i,d}^t) \quad (6)$$

The movement operation or the local search of particles in PSO relies on three categories:

1. Every subsequent move of particles depends on the variable of velocity (which indicates the movement of particles is not an arbitrary or randomly).
2. The local information that comes from the local best solution variable, which interacts with the chosen particle's next move value.
3. The global best solution variable, which also has a great effect on the particle's next move.

The ABC algorithms has none of all aforementioned. The solution improvement or local search of bee relies on the size determined randomly from a random selected neighbour solution, as in Eq. (2). This shows that all add value are completely random on the experimental solutions in every single loop, whereas the PSO has the velocity variable. The local search of ABC algorithm lacks to an information-based procedure, where the discovered solutions (the best) usually not held in the population. Therefore, it will be exchanged with other generated solutions randomly by scout bee; its might not clearly contribute in generate experimental solutions. The improvement part (local search) of employed bee and onlooker bee stage, focusing only on the solution will be selected. However, the higher fitness value of the solutions used to produce experimental solutions in onlooker bee. This means the onlooker bee select the solution depends on the probability value of the solution. On the other hand, scout bee provides the mechanism of global search of ABC with the capability to curb the search from the premature convergence problems. This ability of premature convergence is not existed into PSO search procedures. However, the best solution found usually kept in the population of PSO, and later with new velocities can be used for producing new solutions. This illustrates that the regions concerned of search space are inspected in detail and shorter time, whereas may be the reason that the algorithm will be trapped in the local minima, for this reason the "limit parameter" of the ABC is existed in the proposed HABC. The limit parameter of ABC prevents the algorithm from trapped in the local minima by inserting into the search space a random selected solution from time to time. The proposed HABC algorithm, we combined the advantages of the PSO with the advantages of ABC to solve the optimization search problems. Nevertheless, still there are considerable controversies between researchers about ABC and PSO, where some of them consider PSO faster than the ABC in terms of the execution time while ABC more efficient than the PSO in terms of the accuracy, for some optimization problems [23].

In the proposed HABC algorithm, the colony size of bees composed of the employed, onlookers and scouts bee. Therefore, both food source and colony size are equal to each other. The employed bee of an abandoned food source becomes a scout, then the discovered new source by scout bee, will replace with the abandoned food source. The search carried out by the HABC illustrated as follows:

- The first step of ABC algorithm starts search for a food source by employed bee, then determine the nectar amounts of the source.
- Share the information of the determined food source by the employed bee in the search space with onlooker bee inside the hive (within dance area), and choose the food sources with higher nectar by the onlooker bee.
- Then being investigating of the selected food source to find new food source using Eq. (6).

A few of employed bee becomes a scout bee for the food source have been abandoned, and begins randomly search for a new food source using Eq. (1).

The HABC algorithm main steps are shown as follows in Fig 2:

1: *Initialization step: The same process as the original ABC and PSO algorithms.*
2: *REPEAT*
3: *Move the employed bees onto their food sources and determine their nectar amounts.*
4: *Calculate the probability value of the sources with which they are preferred by the onlooker's bee.*
5: *Move the Onlookers onto the food sources and determine their nectar amounts.*
6: *Move the scouts to search for new food sources replacing the abandoned ones.*
7: *Memorize the best food source found so far.*
8: *UNTIL (requirements are met).*

Fig. 2 Hybrid Artificial Bee Colony algorithm

The HABC algorithm search cycle composed of three steps: in the first, calculate the food source probability in the search space by the employed bee then select food source with the higher fitness randomly to be searched and select by the onlooker bees. Second, regarding the local search of the employed bee phase. Eq. (6) of PSO algorithm will be used instead of Eq. (2). Third, search a new food source by scout bee for those how exceed the limit parameter (similar to the original ABC), using Eq. (1).

IV. COMBINATORIAL INTERACTION TESTING

Theoretical background

The main motivation of t-way testing is to produce numbers of a test case called test suite. These test suites are an array (n x m). Where n indicates the row's number of produced test cases (each test case represents a combination of value (m)). A test suite covers all input parameters and values, where many couples of values can be covered by only one test case. The main problem of t-way testing is to find the most effective test suite that has the small number of the test cases (number of rows).

Generally, test suite involves a number of parameters (*P1, P2, ..., Pn*), each parameters has specific number of values (*V1, V2, ..., Vi*) and the interaction strength t of test suite is an *N x n* array, where each column has only one component of values and the sub-array *N x t* involves all combination of t at least once time only.

Combinatorial testing theoretically is relied on the popular mathematical theme namely Covering Array (CA) to generate the test suite [24], where CA have received more attention as an efficient alternative to the Orthogonal Array (OA) (the oldest mathematical theme), which used for statistical experiments [25]. In general, each system under test (SUT) consists of various elements such as parameters (*P*) and their values (*V*) (where *P* and *V* represent the number of parameters and value respectively). In addition to the *P* and *V*, there is *t* denote the interaction strength level.

Definition 1: As mentioned earlier, each SUT has number of parameters (*P*) associated with their value (*V*), when these *V* are equal to each other for all *P*, then the CA called uniform interaction strength CA(N, *t*, V^P). For instance, let us assume there is a system with four parameters associated with two values for each parameter can be represented as CA (6; 2, 2^4). The system involves of six test cases (rows) that produced based on four parameters (columns).

Definition 2: Mixed covering array is quite the contrary of uniform interaction strength (where the number of values are different for each parameters) can be represented as MCA (N, *t*, $v1^{p1}$ $v2^{p2}$ $v3^{p3}$ . . …vi^{pj}). For instance, system with four parameters (3-parameters have 2-values and 1-parameter have 3-values) can be represented as MCA (12, 3, 2^3 3^1). The system involves of 12 test cases (rows) that produced based on four parameters (columns).

T-way test generation problem

Generating test cases relies on the *t*-way testing technique, these techniques in turn depends on the interaction element's behavior inside the system. To explain the concept of *t*-way testing techniques in generating the optimal test suite, we consider the display tab of a file as a simple example for the basis of our problem as following in Figure 3. Figure 3 shows a screenshot of a display's tab for the file. The display's tab consists of four groups of features that have one or more variable or values; the Navigation pane group, preview pane, details pane group; layout group and sort by group. The display's tab of file provides simple wide levels and factors (i.e., called parameters and values). The display tab consists of four parameters; one 4-value parameters (i.e. Navigation pane), two 2-value parameters (i.e. preview pane and details pane), one 8-value parameters (i.e. layout group) and one 9-value parameters (i.e. sort by). The interaction strength for display tab is assumed *t*=2; the CA is represented as MCA (N, 2, 4^1 2^2 8^1 9^1).

The exhaustive test case to test this tab fully is ($4^1 \times 2^2 \times 8^1 \times 9^1$ = 1152) test cases. In case the interaction strength for this tab is *t*=2, the generated test suite has 72 test case only. Therefore, we are saving around 80% of money and time. However, if there is an increase of interactions between the parameters, then the number of the test cases will be

increasing too. Generally, each combination of value is covered at least once by one test case [4, 26]. NASA application studies showed if only one parameter (*t* = 1) is tested at least once, where can detect 67% of failures, if each pair of parameters (*t* = 2) is tested, can detect 93% of failures, and can detect 98% of failures if every three elements interact is tested. Additionally, the rate of fault detection can achieves 100% if the (*t* = 4-6) interaction of elements [27, 28].

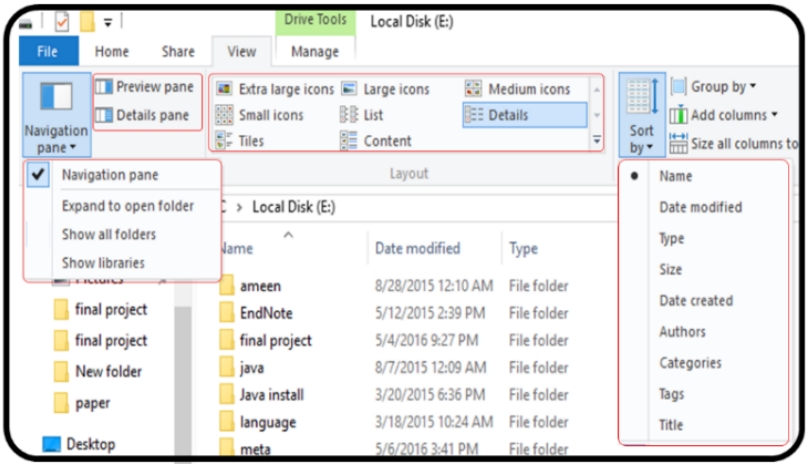

Fig. 3 View of the folder

V. TUNING OF HABC PARAMETERS

In order to achieve a best performance for the proposed HABC strategy, there is a need to the parameters tuning. The performance of the HABC strategy largely is depended on the number of colony size (*NBees*), number of food source, limit, number of cycles (*maxCycle*), learning factor (*C1*and *C2*) and weight factor (*w*).

To tune the control parameters, we adopt a covering array of CA (N; 2, 5^7) as a case study. The verity behind choosing this configuration, many strategies of the *t*-way testing in the literature have been used this covering array for tuning. To realize statistical significance, HABC strategy carried out twenty times with each parameter value, the best and the average values have been recorded from results. We have adopted several values, where the number of bee range are (4-10) and (20-100) and the number of cycle (10, 20, 30, 40, 50, 60, 70, 80, 90, 10, 200, 300, 400, 500, 600, 700, 800, 900, 1000 and 2000). Regarding to the rest of parameters, we used fixed values for both of limit= 100, learning factor (*C1*and *C2*) = 2.0 and weight factor (*w*) = 0.9. Table 2 and 3 presents the generated values of the test suite size (best values and average values) for twenty times executions.

Table. 2 Best and Average Test Suite for CA (N; 2, 5^7)

No. of Bee	Number of Cycle																			
	10		20		30		40		50		60		70		80		90		100	
	Best	Avg.	Best	Avg.	Best	Avg.	Best	Avg.	Best	Avg.	Best	Avg.	Best	Avg.	Best	Avg.	Best	Avg.	Best	Avg.
4	45	46.65	42	44.55	42	43.50	41	42.90	40	42.50	40	42.00	40	41.40	40	41.40	39	40.9	39	40.55
5	45	47.50	42	44.40	42	43.15	41	42.70	41	42.55	41	42.15	39	41.60	39	41.60	40	41.25	39	41.00
6	45	47.85	44	45.15	43	44.45	41	42.75	41	42.25	41	42.70	40	42.15	41	41.75	40	41.55	39	41.30
7	45	47.75	44	45.40	43	44.35	40	43.05	41	43.05	40	41.85	41	42.20	40	41.65	40	41.40	39	41.50
8	45	47.35	43	45.00	42	43.95	41	43.05	41	42.85	41	42.05	41	41.95	40	41.75	40	41.25	40	41.25
9	46	47.55	44	45.40	43	44.25	42	42.90	42	42.90	41	42.30	41	42.10	41	42.30	40	41.85	40	41.60
10	46	47.85	44	45.90	42	44.10	42	43.45	42	42.95	41	42.50	41	42.30	39	41.85	40	41.60	40	41.55
20	44	47.30	44	45.40	42	44.65	42	43.80	41	43.20	41	42.85	40	42.35	41	42.35	40	42.05	40	41.85
30	45	46.55	44	45.55	44	44.70	42	44.25	42	43.40	40	42.70	41	42.90	39	42.30	41	42.15	40	41.70
40	43	46.05	44	45.15	43	44.05	42	43.95	41	43.10	42	43.40	41	42.20	41	42.50	40	42.00	40	41.75
50	44	45.55	42	44.35	42	43.50	41	43.30	41	43.05	40	42.70	40	42.35	40	42.15	41	42.20	40	41.95
60	43	44.95	43	44.30	42	43.30	42	43.45	42	43.15	42	43.00	41	42.35	41	42.05	40	42.10	40	41.40
70	43	45.20	43	44.40	42	43.70	42	43.20	40	42.50	40	42.40	40	42.20	40	42.05	41	42.05	39	41.90
80	42	44.55	42	44.10	41	43.15	42	43.20	41	42.30	41	42.90	40	42.10	40	41.90	40	41.75	40	41.55

90	43	44.25	42	43.75	42	43.20	41	42.85	42	42.90	40	42.55	40	41.95	40	42.40	40	41.50	40	41.45
100	43	44.00	42	43.45	40	42.80	40	42.50	41	42.60	40	42.10	40	41.90	39	41.35	40	41.70	40	41.25

Table. 3 Best and Average Test Suite for CA (N; 2, 5^7)

No. of Bee	Number of Cycle																			
	200		300		400		500		600		700		800		900		1000		2000	
	Best	Avg.	Best	Avg.	Best	Avg.	Best	Avg.	Best	Avg.	Best	Avg.	Best	Avg.	Best	Avg.	Best	Avg.	Best	Avg.
4	38	39.95	37	39.00	37	38.85	38	38.85	37	38.50	37	37.85	36	38.10	37	38.35	37	38.15	36	37.90
5	37	39.90	38	39.05	37	38.75	37	38.60	37	38.70	37	38.35	36	38.05	37	38.00	**35**	38.25	36	37.60
6	39	40.10	39	39.60	38	39.20	37	39.15	37	38.80	38	38.70	36	38.60	38	39.20	37	38.50	36	38.70
7	39	40.10	38	39.75	37	39.45	38	39.05	36	38.80	37	38.70	37	38.30	37	38.25	36	38.35	36	37.85
8	39	40.70	38	39.95	38	39.40	38	39.25	38	39.05	36	38.70	37	38.55	37	38.45	**35**	38.80	36	38.05
9	39	40.40	38	39.80	38	39.20	38	39.05	38	38.80	38	38.90	37	38.70	37	38.75	37	38.40	37	38.45
10	38	40.10	38	40.00	38	39.60	37	39.45	38	39.30	36	39.10	37	38.65	37	38.40	37	38.65	36	37.90
20	39	40.35	38	39.70	38	39.50	37	39.80	37	39.30	38	38.85	37	38.50	37	38.55	37	38.55	37	38.60
30	39	40.55	38	39.85	39	40.15	37	39.15	38	39.10	38	39.25	38	38.80	37	38.65	37	38.85	36	38.05
40	39	40.60	38	39.90	38	39.70	37	39.35	37	38.80	38	39.50	37	38.55	37	38.85	37	38.60	36	37.75
50	38	40.30	38	39.95	37	39.55	38	39.65	37	39.35	38	39.25	37	39.05	37	38.65	36	38.35	37	38.10
60	39	40.60	38	39.80	38	39.60	38	38.95	38	35.50	38	39.15	37	38.85	37	38.85	36	38.30	37	38.05
70	39	40.30	38	39.70	38	39.50	38	39.10	37	39.05	37	38.70	38	38.80	37	38.65	36	38.50	37	38.75
80	40	40.55	38	39.80	38	39.65	37	39.25	37	38.80	37	38.70	37	38.55	37	38.50	36	38.35	37	38.45
90	39	40.60	38	39.95	38	39.20	37	39.15	38	39.30	36	38.90	37	39.05	37	38.80	36	38.80	37	38.40
100	39	40.40	38	39.80	38	39.50	37	39.35	37	39.30	37	39.10	37	38.85	37	38.55	37	38.55	37	38.10

As shown in Table 2 and 3, clearly increasing the number of cycle (maxCycle) can produce a good result. The HABC strategy produces poor outcomes (test suite size) when the number of cycle (maxCycle) < 100. On the other hand, the best outcomes obtained when the number of cycle (maxCycle) more than 100 by simulations. In terms of the number of colony size (NBees), the outcomes showed that there is a positive correlation between the number of colony size and the test suite size. Improving the HABC strategy performance by increased the number of colony size (NBees). Finally, the best test suite that have been recorded when the number of colony size (NBees) = 8 and the number of cycle (maxCycle) = 1000. In sum, the HABC strategy generates the best test suit size when the number of cycle greater than 700.

VI. RESULT

The experiments goal is to investigate the performance of the tuned parameters for HABC strategy, where adpoted this experiments that collected from [2, 4, 16, 29]. In order to ensure significant comparison, the HABC strategy have compared with the pure computational based strategies and meta-heuristics based strategies based on the adopted parameters value for number of colony size = 5 and number of cycle = 1000.

Table 4. Shows a comparison the existing *t*-way strategies with HABC strategy in order to know the performance of HABC. The cell with NA signified "Not Available results." the shaded cells are portrayed the best test suite, and the cell marked with * represents the optimal test suite. HABC strategy shows the ability to produce the most minimum test suite size for Configuration number (1, 5, 7, 8, 12 and 13) and the optimal test suite size for CA(N; 2, 5^{10}). However, still HABC strategy not exceed the some strategies; but the results are still better and sufficiently competitive comparing to existing strategies.

VII. CONCLUSION

This paper have proposed a new *t*-way testing strategy based on the hybrid artificial bee colony algorithm called HABC strategy with the parameters tuning. The experiment result was encouraging, where HABC performance outperformed the other existing strategies for some configurations and produced the most minimum for other configurations. As part of our future work, to support for high interaction parameters and improving HABC to support the constraints.

Table. 4 Benchmarking HABC with existing strategies

NO.	Configuration	mAETG	AETG	IPOG	Jenny	TVG	SA	ACA	GA	PSO	HSS	CSS	HABC
1	CA(N; 2, 3^4)	9	9	9	10	11	9	9	9	9	9	9	9
2	CA(N; 2, 3^{13})	17	15*	20	20	19	16	17	17	17	18	18	19
3	CA(N; 2, 10^{10})	NA	NA	176	157	208	NA	159	157	NA	155	151*	186
4	CA(N; 2, 15^{10})	NA	NA	373	336	473	NA	NA	NA	NA	342	333*	417
5	CA(N; 2, 5^{10})	NA	NA	50	45	51	NA	NA	NA	45	43	42	40*
6	CA(N; 3, 3^6)	38	47	53	51	49	33	33	33	42	39	40	45
7	CA(N; 3, 4^6)	77	105	64	112	123	64	64	64	102	70	101	64
8	CA(N; 3, 5^6)	194	NA	216	215	234	152	125	125	NA	199	197	125
9	CA(N; 3, 6^6)	330	343	382	373	407	300*	330	331	338	336	334	337
10	CA(N; 3, 5^7)	218	229	274	236	271	201*	218	218	229	236	218	221
11	CA(N; 3, 10^6)	1426*	1496	NA	1572	1717	1508	1501	1473	1506	1505	1470	1490
12	MCA(N; 2, 5^1 3^8 2^2)	20	19	19	23	22	15	16	15	NA	20	20	15
13	MCA(N; 2, 7^1 6^1 5^1 4^6 3^8 2^3)	44	45	43	50	51	42	42	42	48	50	47	42
14	MCA(N; 3, 5^2 4^2 3^2)	114	NA	111	131	136	100	106	108	NA	120	121	113
15	MCA(N; 3, 10^1 6^2 4^3 3^1)	377	NA	383	399	414	360	361	360	385	378	386	371

REFERENCES

1. B. S. Ahmed, L. M. Gambardella, W. Afzal, and K. Z. Zamli, "Handling constraints in combinatorial interaction testing in the presence of multi objective particle swarm and multithreading," Information and Software Technology, vol. 86, pp. 20-36, 2017.
2. 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.
3. X. Chen, Q. Gu, A. Li, and D. Chen, "Variable strength interaction testing with an ant colony system approach," in Software Engineering Conference, 2009. APSEC'09. Asia-Pacific, 2009, pp. 160-167.
4. 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.
5. Nasser, Y. A. Alsariera, K. Z. Zamlifll, and B. Al—Kazcmi, "Late acceptance hill climbing based strategy for addressing constraints within combinatorial test data generation," 2014.
6. K. Alazzawi, H. M. Rais, and S. Basri, "Artificial Bee Colony Algorithm for t-Way Test Suite Generation," in 2018 4th International Conference on Computer and Information Sciences (ICCOINS), 2018, pp. 1-6.
7. A. A. Alsewari, A. K. Alazzawi, T. H. Rassem, M. N. Kabir, A. A. B. Homaid, Y. A. Alsariera, et al., "ABC Algorithm for Combinatorial Testing Problem," Journal of Telecommunication, Electronic and Computer Engineering (JTEC), vol. 9, pp. 85-88, 2017.
8. K. Alazzawi, A. A. B. Homaid, A. A. Alomoush, and A. A. Alsewari, "Artificial Bee Colony Algorithm for Pairwise Test Generation," Journal of Telecommunication, Electronic and Computer Engineering (JTEC), vol. 9, pp. 103-108, 2017.
9. Y. A. Alsariera and K. Z. Zamli, "A real-world test suite generation using the bat-inspired t-way strategy," presented at the In the 10th Asia Software Testing Conference (SOFTEC2017), 2017.
10. Y. A. Alsariera, H. S. Alamri, and K. Z. Zamli, "A Bat-Inspired Testing Strategy for Generating Constraints Pairwise Test Suite," presented at the The 5th International Conference on Software Engineering & Computer Systems (ICSECS), 2017.
11. Y. A. Alsariera, A. Nasser, and K. Z. Zamli, "Benchmarking of Bat-inspired interaction testing strategy," International Journal of Computer Science and Information Engineering (IJCSIE), vol. 7, pp. 71-79, 2016.
12. Y. A. Alsariera and K. Z. Zamli, "A bat-inspired strategy for t-way interaction testing," Advanced Science Letters, vol. 21, pp. 2281-2284, 2015.
13. Y. A. Alsariera, M. A. Majid, and K. Z. Zamli, "Adopting the bat-inspired algorithm for interaction testing," presented at the The 8th edition of annual conference for software testing, 2015.
14. Y. A. Alsariera, M. A. Majid, and K. Z. Zamli, "SPLBA: An interaction strategy for testing software product lines using the Bat-inspired algorithm," in Software Engineering and Computer Systems (ICSECS), 2015 4th International Conference on, 2015, pp. 148-153.
15. Y. A. Alsariera, M. A. Majid, and K. Z. Zamli, "A bat-inspired Strategy for Pairwise Testing," ARPN Journal of Engineering and Applied Sciences, vol. 10, pp. 8500-8506, 2015.
16. J. Stardom, Metaheuristics and the search for covering and packing arrays: Simon Fraser University, 2001.
17. R. C. Eberhart and J. Kennedy, "A new optimizer using particle swarm theory," in Proceedings of the sixth international symposium on micro machine and human science, 1995, pp. 39-43.
18. D. Karaboga, "An idea based on honey bee swarm for numerical optimization," Technical report-tr06, Erciyes university, engineering faculty, computer engineering department2005.
19. K. M. Passino, "Biomimicry of bacterial foraging for distributed optimization and control," IEEE control systems, vol. 22, pp. 52-67, 2002.
20. D. Karaboga and B. Akay, "A survey: algorithms simulating bee swarm intelligence," Artificial Intelligence Review, vol. 31, pp. 61-85, 2009.
21. X. Yan, Y. Zhu, and W. Zou, "A hybrid artificial bee colony algorithm for numerical function optimization," in Hybrid Intelligent Systems (HIS), 2011 11th International Conference on, 2011, pp. 127-132.
22. M. S. Kıran and M. Gündüz, "A novel artificial bee colony-based algorithm for solving the numerical optimization problems," International Journal of Innovative Computing, Information & Control, vol. 8, pp. 6107-6121, 2012.
23. Z. N. Alqattan and R. Abdullah, "A comparison between artificial bee colony and particle swarm optimization algorithms for protein structure prediction problem," in International Conference on Neural Information Processing, 2013, pp. 331-340.

24. Yilmaz, M. B. Cohen, and A. A. Porter, "Covering arrays for efficient fault characterization in complex configuration spaces," IEEE Transactions on Software Engineering, vol. 32, pp. 20-34, 2006.
25. M. B. Cohen, P. B. Gibbons, W. B. Mugridge, and C. J. Colbourn, "Constructing test suites for interaction testing," in Software Engineering, 2003. Proceedings. 25th International Conference on, 2003, pp. 38-48.
26. M. Cohen, S. R. Dalal, J. Parelius, and G. C. Patton, "The combinatorial design approach to automatic test generation," IEEE software, vol. 13, pp. 83-88, 1996.
27. A W. Williams, Software component interaction testing: Coverage measurement and generation of configurations: University of Ottawa (Canada), 2002.
28. K. Z. Bell and M. A. Vouk, "On effectiveness of pairwise methodology for testing network-centric software," in Information and Communications Technology, 2005. Enabling Technologies for the New Knowledge Society: ITI 3rd International Conference on, 2005, pp. 221-235.
29. K. J. Nurmela, "Upper bounds for covering arrays by tabu search," Discrete applied mathematics, vol. 138, pp. 143-152, 2004.