Combinatorial Test Techniques: Artificial Bee Colony Algorithm for Test Suite Generation

By

Shrabanti Saha Rimi
(162-35-1649)

A Thesis submitted in partial fulfillment of the requirement for the degree of Bachelor of Science in Software Engineering

Supervised By

Mr. Md. Rashed Karim
Assistant Professor
Department of Software Engineering
Daffodil International University

Department of Software Engineering
DAFFODIL INTERNATIONAL UNIVERSITY

Spring – 2020

This thesis titled on “Combinatorial Test Techniques: Artificial Bee Colony Algorithm for Test Suite Generation”, submitted by Name: Shrabanti Saha Rimi,
ID:162-35-1649 to the Department of Software Engineering, Daffodil International University has been accepted as satisfactory for the partial fulfillment of the requirements for the degree of Bachelor of Science in Software Engineering and approval as to its style and contents.

BOARD OF EXAMINERS

Declaration

I hereby declare that, this thesis has been done by me under the supervision of Mr. Md. Rashed Karim, Assistant Professor, and Department of Software Engineering, Daffodil International University. I also declare that neither this thesis nor any part of this thesis has been submitted elsewhere.

Certified By:

_____

Mr. Md. Rashed Karim.
Assistant Professor.
Department of Software Engineering
Daffodil International University

Submitted By:

_____

Shrabanti Saha
Rimi 162-35-1649
Department of Software Engineering
Daffodil International University

Acknowledgement

First of all, I would like to express my gratitude to Almighty for enabling me to complete this report on “Combinatorial Test Techniques: Artificial Bee Colony Algorithm for Test Suite Generation”.

I convey my sincere gratitude to my Academic Supervisor Mr. Md. Rashed Karim, Assistant Professor of Department of SWE, Daffodil International University. Without his caring bearing and appropriate direction, this learning would have been a little achievement.
In every phase of the thesis, his supervision and guidance shaped this report to be completed perfectly.
Next to him are my parents, whom I am extraordinarily obligated for me raised with affection and support to this stage.

At last but not least I am thankful to all my teachers and friends who have been always helping and encouraging me throughout the year. I have no valuable words to express my thanks, but my heart is still full of the favors received from every person.

TABLE OF CONTENTS **PAGE**

LIST OF FIGURES

LIST OF TABLES

LIST OF ABBREVIATIONS

ABC	Artificial Bee Colony
PSO	Particle Swarm Optimization
CTT	Combinatorial Testing Technique
DE	Development Engineer
EA	Electronic Arts
ACS	Ant Colony System
SA	Simulated Annealing
HSS	Harmony Search Strategy
TLBO	Teacher-Learning-based Optimization

ABSTRACT

Artificial Bee Colony (ABC) is another populace based stochastic calculation that has demonstrated great inquiry capacities on numerous streamlining issues. The fundamental errand in executing t-way testing procedures is building the best chance experiment. There are a few strategies that have been proposed yet none of them can be professed to be the best outcome since t-way is considered as ABC issue. In this paper, the idea of (ABC) will be proposed as the thought in building t-way testing test suit. Because of the impediment of old-style streamlining in unraveling bigger scope combinatorial and exceptionally non-straight issues, specialists are moving to utilize the astute conduct of swarm known as Multitude Knowledge. ABC was presented by Karaboga in 2005 is an improvement calculation dependent on bumblebee swarm which has been applied in explaining true application. Pairwise testing is a combinatorial procedure used to lessen the number of experiment contributions to a framework in circumstances where comprehensive testing with all conceivable inputs isn't achievable. This paper presents the effects of producing pairwise test sets utilizing an (ABC) province calculation. Contrasted with distributed outcomes for sphere benchmark issues, the (ABC) province approach created test sets that were or better as far as size for each of the iteration issues. The outcomes show that the age of pairwise test sets utilizing an (ABC) bee province calculation is conceivable, and recommend that the methodology might be valuable in testing situations where pairwise test set information will be reused and find the optimal result. (ABC) calculation is an improvement calculation dependent on a specific keen conduct of (ABC) swarms. This work contrasts the exhibition of ABC calculation and that of differential development (DE), molecule swarm advancement (PSO), and transformative calculation (EA) for multi-dimensional numeric issues. The reproduction results show that the exhibition of ABC calculation is tantamount to those of the referenced calculations and can be effectively utilized to take care of design issues with high dimensionality. Combinatorial Testing Technique (CTT) is one of the famous techniques that have been used in fault detection of the software systems. Pairwise testing is one of the efficient CTT that used widely for fault detection based on the caused failures by two interaction parameters. Many researchers have been developed as a pairwise testing strategy. Supplementing to the previous explores, this paper proposes another pairwise test suite age called Pairwise test (ABC) methodology dependent on hybridizing of a Province (ABC) calculation with a Molecule Multitude Improvement (PSO) calculation. Empirical results show that the ABC strategy outperforms other strategies in some cases and provides competitive results in other cases by generating the final test suite.

CHAPTER 1

INTRODUCTION

1.1 Background

Artificial bee colony algorithm is a metaheuristic algorithm, which was proposed by Karaboga in 2005. It is a swarm intelligence algorithm swarm intelligence has been formally defined as any attempt to design algorithms or distributed problem-solving devices which are inspired by the behavior of social insect colonies or other animal societies.

Our life increasingly relies on the software along with the advancement of technology such as the communications (e.g. Skype), businesses (e.g. Marketing & Sales), education (e.g. Math Problem Solving), Navigation (e.g. Waze), etc. Software use is unavoidable. Software development leads to the creation of a software application with more complexity. Therefore, the expectation of faults that may occur because of an increase to the very large number of combinations among hundreds of parameters (inputs), will grow [Yilmaz, C., Fouche, S., Cohen, M. B., Porter, A., Demiroz, G., & Koc, U. (2013)]. Due to the large number of inputs as well as time consumption for a test and cost, it is becoming hard or impossible to get exhaustive testing for any software to fault detection. For this reason, the Combinatorial Testing Technique (CTT) is one of the famous techniques that have been used in fault detection of the software systems. A special case of the combinatorial testing technique is pairwise testing. Pairwise testing is one of the efficient and effective Combinatorial Testing Techniques (CTT) for feasible solutions, which is based on the caused failures by two interactions parameters value of the configuration system (inputs) [Hervieu, A., Marijan, D., Gotlieb, A., & Baudry, B. (2016), Kuhn, D. R., Bryce, R., Duan, F., Ghandehari, L. S., Lei, Y., & Kacker, R. N.

(2015)]. The main purpose of using pairwise testing strategies because most of the failures as a result of interaction can be caused by two parameters at most depending on investigation [Nasser, A. B., Sariera, Y. A., Alsewari, A. R. A., & Zamli, K. Z. (2015), Kuhn, D. R., & Reilly, M. J. (2002, December)]. There are many researches have reported that around 75% of errors can be discovered by pairwise testing [Kuhn, D. R., Wallace, D. R., & Gallo, A. M. (2004)].

Producing the minimum test cases (Which indicates test suite size) that can cover all the parameter's interaction value is still a Nondeterministic Polynomial (NP) hard problem [Colbourn, C. J., Cohen, M. B., & Turban, R. (2004, February), Khalsa, S. K., & Labiche, Y. (2014, November), Al-Sewari, A. A., & Zamli, K. Z. (2014), Nie, C., & Leung, H. (2011)]. Therefore, CTT strategies are used to decrease the number of test cases and generate the best test suite that can cover all possible combinations. Many existing strategies in the literature have been suggested by researchers to decrease the number of test cases and produce the best result that can cover all possible combinations [Alsariera, Y. A., Ahmed, H. A. S., Alamri, H. S., Majid, M. A., & Zamli, K. Z. (2018), Alsariera, Y. A., Nasser, A., & Zamli, K. Z. (2016), Alsariera, Y. A., & Zamli, K. Z. (2015), Alsariera, Y. A., Majid, M. A., & Zamli, K. Z. (2015), Alsariera, Y. A., & Zamli, K. Z. (2015)]. These strategies are designed based on the AI-algorithms to solve the optimization problems to produce the optimal solution such as Genetic Algorithm (GA) [Flores, P., & Cheon, Y. (2011, June), Shiba, T., Tsuchiya, T., & Kikuno, T. (2004, September)], Harmony Search (HS) [Alsewari, A. R. A., & Zamli, K. Z. (2012)], Bat Algorithm (BA) [Alsariera, Y. A., & Zamli, K. Z. (2015)], Simulated Annealing (SA) [Cohen, M. B., Gibbons, P. B., Mugridge, W. B., Colbourn, C. J., & Collofello, J. S. (2003, November)], Cuckoo Search Algorithm (CS) [Ahmed, B. S., Abdulsamad, T. S., & Potrus, M. Y. (2015)], Ant Colony Algorithm (ACA) [Shiba, T., Tsuchiya, T., & Kikuno, T. (2004, September)], Particle Swarm (PS) [Rabbi, K., Mamun, Q., & Islam, M. R. (2015, June)], Artificial Bee Colony (ABC) [Alazzawi, A. K., Rais, H. M., & Basri, S. (2018, August), Alazzawi, A. K., Rais, H. M., & Basri, S. (2019, April), Alazzawi, A. K., Homaid, A. A. B., Alomoush, A. A., & Alsewari, A. A. (2017)], Kindly algorithm (KA) [Homaid, A. A. B., Alsewari, A. A., Alazzawi, A. K., & Zamli, K. Z. (2018)], Flower

Pollination Algorithm (FPA) [Nasser, A. B., Alsewari, A. A., Tairan, N. M., & Zamli, K. Z. (2017)], Teaching Learning-based Optimization (TLBO) [Zamli, K. Z., Din, F., Baharom, S., & Ahmed, B. S. (2017)], etc. Most of these strategies produced the most optimal test suite size better than other strategies that were based on the greedy, mathematic, and random algorithms. Therewith, AI-algorithms need to be run more than one time to produce the optimal solution [Khalsa, S. K., & Labiche, Y. (2014, November), Nie, C., & Leung, H. (2011)].

So, one other swarm intelligence algorithm which we have so far seen is particle swarm optimization. This is also a swami intelligence algorithm examples of swarms are bees swarming around their hive, ant colony with ants as individual agents, the flock of birds it is a swarm of birds, there mean the system is a swarm of cells and crowd is a swarm of people. These are examples of swarms. So, any algorithm which is designed with this as its inspiration is a swarm intelligence algorithm. Again these are only examples there can be many other examples. So, the two properties of swarm intelligent behavior are self-organization and division of labor by the self-organization, we mean that the interactions are executed based on purely local information without any relation to the global pattern. So, the four components of self-organization are positive feedback, negative feedback, fluctuations, and multiple interactions. By the division of labor, we mean that the task that need to be performed are simultaneously performed by specialized individuals. So, particles form optimization which we have previously was modeled based on the social behavior of bird flocking or fish cooling.

So, an artificial bee colony algorithm was first proposed in 2005 by Dervis Karaboga. Subsequently, it was published in 2007 in the Journal of Global Optimization. So, ever since its publication, it has been receiving increased interest from researchers both in terms of its applications as well as variants of ABC algorithm that have been proposed. In 2015, Dervis Karaboga coauthored a publication in which they clarified some of the

misconceptions concerning artificial bee colony algorithm. Like all other algorithms; ABC has found applications in engineering social sciences and also in business management.

1.2 Motivation of The Research

A few present-day heuristic calculations have been created for taking care of combinatorial and numeric advancement issues. ABC calculations can be arranged into various gatherings relying upon the rules being thought of, for example, populace based, iterative based, stochastic, deterministic, and so forth. While a calculation working with a lot of arrangements and attempting to improve them is called populace based, the one utilizing different emphases to move toward the arrangement looked for is named as an iterative calculation. For this reason, various t-way interaction-testing strategies for the past 20 years have been propounded in works of literature. All the planned methodologies help with building test suites with insignificant experiments to accomplish all the communication quality proposed. According to this purpose, we have to motivated that develop a combinatorial testing strategy Based on the ABC algorithm.

1.3 Problem Statement

The Counterfeit Honey Bee Province Calculation has been concentrated as the rudimentary aptitude for the t-way test age. Introducing other substitute optimization algorithms except for ACS, SA, HSS, and PSO might lead to new views in generating the t-way test. Many combinations of input parameters, hardware configurations, software configurations, and system platforms need to be tested and verified against the given requirements. To test all possible combinations, exhaustive testing is perhaps the most effective testing strategy. However, due to time and cost constraints, exhaustive testing is impractical and not cost-efficient [Bertolino, A. (2007, May)]. Thus, many sampling strategies such as boundary value, decision table, and equivalence partitioning are used in

fault detection and prevention. In any case, the principle issue with this static and dynamic testing is that the things are not compelling to handle bugs because of collaboration [Othman, R. R., Zamli, K. Z., & Mohamad, S. M. S. (2013).]. As a better option, t-way strategies have been developed in the literature in the last 20 years adopting different approaches such as Algebraic, purely computational, and optimization algorithm. Alsewari and Zamli have reviewed and indicated that the current improvement calculation despite everything has its impediments for every technique, along these lines recommending another test thought and test suite developments. [Al-Sewari, A. A., & Zamli, K. Z. (2014).] Driven by this test, this paper proposes a counterfeit honey bee settlement (ABC) based t-way technique. ABC is claimed to be one of the good search abilities on many, optimization. [Xu, Y., Fan, P., & Yuan, L. (2013).] A recent study has shown that ABC performs significantly better or at least comparable to other swarm intelligence algorithm. According to this all of the papers there didn't have any clear concept for full ABC algorithm process, how to work a full ABC algorithm, and including there have all other issues, all papers didn't mention that how to apply ABC algorithm for the best solution and where is the position of the best solution, also how many iterations need to find the more best solutions, and didn't mention there have how many coverages are happened for using food sources with iterations.

1.4 Research Questions

Comprehensive testing is unimaginable in light of the fact that the number of experiments can be excessively enormous, in any event, for straightforward programming and equipment items. According to the ABC algorithm, there have lots of function-based possible ways to minimize the way and find the best solution to the existing environment. Let us consider that the existing environment has 50 food sources the according to bees need to know that which food source is the near according to them, if they find that food sources, then the bees have the best solution according to low cost and then they find the best solution position. So, based on the ABC algorithm strategy, there has a way that finding the best result with the coverage of the performance.

The above-mentioned examples highlight the common combinatorial explosion problem. Given limited time and resources, the main research questions are as follows:

1. How to find the best solution and best objective according to there, iterations?
2. How to find the best objective position to minimize the test?
3. How many performances, are covered in the whole test?
4. How to implement the ABC algorithm for finding the optimal results?

1.5 Research Objectives

Our life increasingly relies on the software along with the advancement of technology such as the communications (e.g. Skype), businesses (e.g. Marketing & Sales), education (e.g. Math Problem Solving), Navigation (e.g. Waze), etc. Software use is unavoidable. Software development leads to the creation of a software application with more complexity. Therefore, the expectation of faults that may occur because of an increase to the very large number of combinations among hundreds of parameters (inputs), will grow. Due to the large number of inputs as well as time consumption for a test and cost, it is becoming hard or impossible to get exhaustive testing for any software to fault

detection. For this reason, the Combinatorial Testing Technique (CTT) is one of the famous techniques that have been used in fault detection of the software systems. A special case of the combinatorial testing technique is pairwise testing.

1. To analyze recent research on ABC algorithm based test strategies to identify the best solution and the best position in the test strategy.
2. To implement the ABC algorithm for finding optimal results.
3. To evaluate the performance of coverage in ABC algorithm strategy with many different iteration and generated test suite cases.

1.6 Research Scope

This research focus is to implement and develop the t-way strategy with the ABC algorithm, and generate the optimal result with test suite cases. The test implementation and execution activity involves run the tests. The optimal results are based on there, exit criteria. Then that's is defined during test planning and before test execution started based on there, test cases. At the end of test execution, the test manager checks to see if these have been met and the results are optimal or not. Test closure activities concentrate on making sure that everything is organized and finding the best objective position.

The proposed t-way strategy can be used to generate higher strength t-way test suite. In this research, the design of a new t-way strategy is taken as the main target, which can be used to generate optimum and/or minimum number of test cases. Therefore, time and space complexity can be minimized with the proper number of iteration in the ABC algorithm.

1.7 Thesis Organization

This chapter briefly introduced the research background and some preliminary knowledge about combinatorial testing, t-way test strategy, ABC algorithm. Background, the motivation of the research, the problem statement, research questions, research objective, research scopes as well as research contributions are introduced in this chapter. The remainder of the postulation is sorted out as follows.

Chapter 2 begins with the definitions and examples of some terminologies underlying this research. It presents a literature survey on different combinatorial methods for t-way test strategies considering the ABC algorithm. In the mean-time, the existing t-way strategies are explored deeply on the ABC search techniques.

Chapter 3 presents the design and development of the ABC algorithm in a new strategy for software input testing based on the t-way strategy that considers the combinatorial test technique. It also presents the design and development of searching algorithms and design methodology in order to generate a test suit for the t-way strategy.

Chapter 4 discusses and shows the evaluation of the ABC strategy based on finding the correctness of the generated test cases and show the optimal result. Furthermore, the result performance table with the published and well-known t-way test strategies is done in this chapter.

At long last, Part 5 sums up the accomplishment and constraint of the proposed structure and execution. Part 5 finishes up this exploration work with some proposals for future heading.

CHAPTER 2

LITERATURE REVIEW

Since the development of ABC, it has attracted much attention for its excellent characteristics. In the last decade, different versions of ABCs have been applied to various problems. In this area, we present a concise audit of these ABC calculations [Karaboga, D., & Akay, B. (2009).] introduced a near investigation of ABC. A large set of benchmark functions are tested in the experiments. Results show that the ABC is better than or like those of other populace based calculations with the benefit of utilizing less control boundaries.

In [Gao, W. F., Huang, L. L., Liu, S. Y., & Dai, C. (2015)], a new search pattern called ABC/best/1 is utilized to accelerate the convergence speed. In [Gao, W., & Liu, S. (2011)], ABC/best/1, and another hunt design called ABC/rand/1 are utilized. Moreover, a parameter q is intruded to control the frequency of these two patterns. [Zhu, G., & Kwong, S. (2010)] utilized the search information of the global best solution ($g$best) to guide the search of ABC. Reported results show that the new approach achieves better results than the original ABC algorithm. [Akay, B., & Karaboga, D. (2012)] proposed a modified ABC algorithm, in which two new search patterns, frequency, and magnitude of the perturbation, are employed to improve the convergence rate. Results show that the first ABC calculation can productively explain fundamental and straightforward capacities, while the altered ABC calculation gets promising outcomes on the half breed and complex capacities when contrasted with some best in class calculations. [Banharnsakun, A., Achalakul, T., & Sirinaovakul, B. (2011)] changed the pursuit example of the spectator honey bees, in which the best plausible arrangements found so far are shared comprehensively among the whole multitude. Accordingly, the new applicant arrangements are like the current best arrangement.

Kang et al. [Kang, F., Li, J., & Ma, Z. (2011)] proposed a Rosen brock ABC (RABC)

calculation which consolidates Rosen brock's rotational bearing strategy with the first ABC. There are two elective periods of RABC: the investigation stage acknowledged by ABC and the abuse staged finished by the [Wu, B., Qian, C., Ni, W., & Fan, S. (2012)] consolidated amicability search (HS) and the ABC calculation to develop a crossbreed calculation. Examination results show that the half and half calculation outflanks ABC, HS, and other heuristic calculations. Li et al. [Li, G., Niu, P., & Xiao, X. (2012)] proposed an improved ABC calculation called IABC, in which the best-so-far arrangement, latency weight, and increasing speed coefficients are acquainted with changing the hunt procedure. Moreover, a hybrid ABC algorithm (PS-ABC) based on *g*best-guided ABC (GABC) [Zhu, G., & Kwong, S. (2010)] and I-ABC is proposed. Results show that PS-ABC converges faster than I-ABC and ABC. [Karaboga, D., & Ozturk, C. (2011).] utilized ABC calculation for information grouping. Experiments are conducted on thirteen typical test data sets from UCL Machine Learning Repository. The presentation of ABC is contrasted and PSO and other nine arrangement procedures. Reproduction results exhibit that the ABC calculation can proficiently illuminate information bunching. Zhang et al. [Zhang, C., Ouyang, D., & Ning, J. (2010)] additionally utilized ABC calculation for grouping. Three informational indexes are tried. The presentation of ABC is contrasted and hereditary calculation, reenacted toughening, unthinkable inquiry, ACO, and K-NM-PSO. Results demonstrate the Mathematical Problems in Engineering 3 effectiveness of ABC on clustering. [Karaboga, D., & Ozturk, C. (2010)] applied ABC to explain fluffy bunching. Three data sets including cancer, diabetes, and heart has chosen from the UCI database are tested. Results indicate that the performance of ABC is successful in fuzzy clustering.

The ABC algorithm is usually used to solve unconstrained optimization problems. In [Karaboga, D., & Akay, B. (2011)], investigated the performance of ABC on constrained optimization problems. To handle constraints, Deb’s rules consisting of three simple heuristic rules are employed. [Mezura-Montes, E., & Velez-Koeppel, R. E. (2010, July).] proposed an elitist ABC algorithm for constrained real-parameter optimization, in which the operators used by different types of bees are modified. Additionally, a dynamic

tolerance control mechanism for equality constraints is utilized to facilitate the approach to the feasible region of the search space. [Yeh, W. C., & Hsieh, T. J. (2011)] proposed a penalty-guided ABC algorithm to solve reliability redundancy allocation problems. Sabat et al. [Sabat, S. L., Udgata, S. K., & Abraham, A. (2010)] introduced the use of ABC to remove the little sign identical circuit model boundaries of GaAs metal-broadened semiconductor field impact transistor (MESFT) gadget.

The performance comparison shows that ABC is better than PSO. It is known that the ABC algorithm is good at solving optimization problems over continuous search space. For discrete advancement issues, it is a major test for the ABC calculation. Li et al. [Li, J. Q., Xie, S. X., Pan, Q. K., & Wang, S. (2011)] used a hybrid Pareto-based ABC algorithm to solve flexible job shop scheduling problems. In the new algorithm, each food source is represented by two vectors, that is, the machine assignment and the operation schedule. Moreover, an external Pareto archive set is utilized to record non dominated solutions. In [Kashan, M. H., Nahavandi, N., & Kashan, A. H. (2012)], Kashan et al. planned another ABC calculation called Dis ABC to enhance paired organized issues. Szeto et al. [Szeto, W. Y., Wu, Y., & Ho, S. C. (2011)] proposed an enhanced ABC algorithm to solve the capacitated vehicle routing problem. The presentation of the new methodology is tried on two arrangements of standard benchmark occurrences. Recreation results show that the new calculation outflanks the first ABC and a few other existing calculations. Skillet et al. [Pan, Q. K., Tasgetiren, M. F., Suganthan, P. N., & Chua, T. J. (2011)] presented a discrete ABC algorithm hybridized with a variant of iterated greedy algorithm to solve a permutation flow shop scheduling problem with the total flow time criterion.

CHAPTER 3

RESEARCH METHODOLOGY

The pair-wise testing strategy ABC has found applications in engineering social sciences and also in business management. So, this shows the comparison of TLBO and ABC. Before move on to the algorithm, there needs to first understand what are the components of honey bees, swarms. So, there are three major components; one is the food source, the employed bees, and the unemployed bees. The overall method is discussed and calculated.

3.1 Components of Honey Bee

1.Food Sources.
2.Employed foragers.
3.Unemployed foragers. (Onlooker, Scout)

Food source value depends on its proximity, richness, and ease of extraction and they can be represented with a single quantity known as profitability. Each solution has an objective function value associated with it right. So, the objective function value can be considered as profitability. So, there consider as food sources will be the solutions in optimization. Employed bees are currently associated with the food source, they are exploiting a food source, they contain information on distance profitability and direction from the nest. These employed bees share the information with a certain probability to

other bees right. So, when a bee collects nectar and comes back to the hive to unload it has three options; one can either abandon the food source and thus becomes an uncommitted follower. The second is to dance in the dance area right these bees perform something called as waggle dance. So, the other bees who are unemployed as of now look at this dance and they may choose to follow this particular employed bee to go to that particular food source to collect more nectar right. So, the second option for the bees is to dance recruit other bees, and return to the food source and keep collecting the nectar from the food source. The third option is not to dance, but to continue to forager at the food source. In this case, it does not recruit any other bees, but it continues to forager at the food source and collect the nectar and come back to the hive right. So, that is the employed bee. So, there are two types of unemployed foragers; one is onlookers and the other one is a scout. So, these onlookers watch the waggle dance, they may choose to follow that particular bee and they start searching for a food source; whereas, scouts do not necessarily watch the waggle dance, but starts exploring around the nest spontaneously right. So, we have three components; one is a food source, employed foragers and unemployed foragers.

3.2 Artificial Bee Colony Algorithm(ABC)

1.Employed bee phase.
2.Onlooker bee Phase.
3.Scout bee phase.

In the employed bee phase the employed bees try to identify a better food source than the one they are currently associated with the right. In terms of optimization, there will be generating a new solution using a partner or solution in the employ bee phase and then there will be performing a greedy selection. So, as know greedy selection; in the greedy selection, there will be accepting a new solution if it is better than the current solution.

In the onlooker bee phase, there will be generating a new solution and performing a greedy search, so this is similar to the employed bee phase. Only thing is that in the employed bee phase every bee associated with a particular food source was generating a new solution, here we will select a food source based on a probability related to the nectar amount right. So, it is not like every food source will be used to generate a new solution.

In the scout bee phase, the exhausted food source is abandoned. So, in terms of optimization; there will be discarding a particular solution and there will be generating a new solution.

A major difference between many of the other algorithms and ABC is the use of the term fitness. In most of the algorithms, the term fitness directly corresponded to the objective function value.

3.3 Fitness Evaluation and Greedy Selection

Table 3.3.1: Example

	Objective Function	Fitness Function
S1	10	10
S2	5	5

So, if there are two solutions say S1 and S2 right and if the objective function value of S1 was 10 and S2 was 5 right. So, then the fitness of solution 1 was 10 and the fitness of solution 2 was 5. So, the objective function directly correspondent to the fitness function value. And if there are solving a minimization problem, then solution S2 is better than

solution S1. So, objective function directly corresponded to the fitness function value. That is not the case in artificial bee colony optimization.

Fitness of a solution is evaluated as,

Table 3.3.2: Formula

(3.1)

$fit = \frac{1}{1+f}$	if $f \geq 0$
$fit = 1 + \|F\|$	if $f < 0$

In an artificial bee colony algorithm; the fitness is related to the objective function using the solution. So, if the objective function value is greater than or equal to 0, then the fitness is 1 by 1 plus f and if the objective function value is less than 0, then the fitness is given by 1 plus |f|.

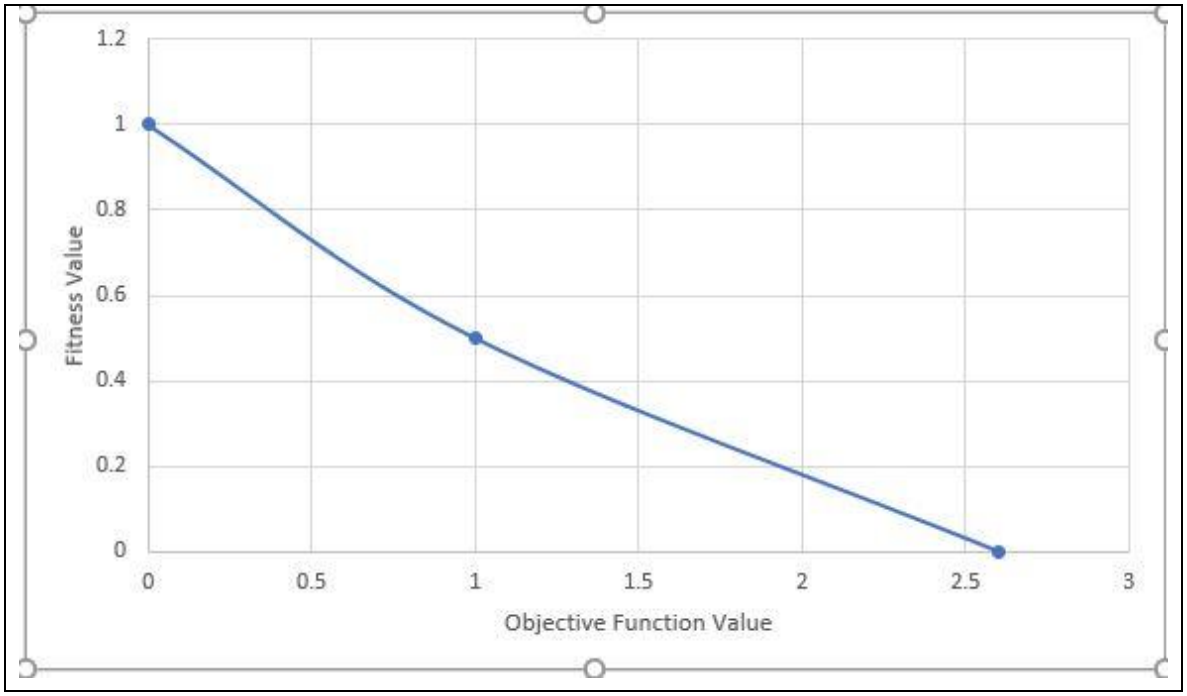

Figure 3.3.1: Formula Graph-1

So, this graph shows the variation of fitness function value on the Y-axis and the objective function value on the X-axis. There can see as the objective function value increases; the fitness value decreases.

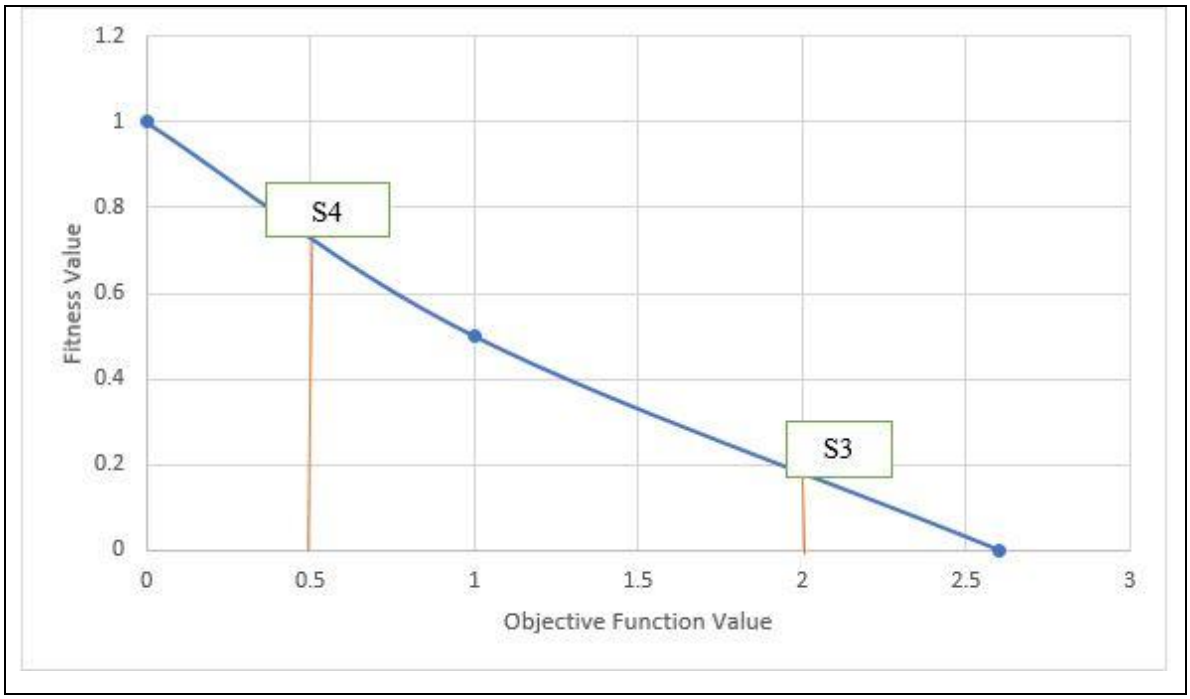

Figure 3.3.2: Formula Graph-2

So, if there consider two solutions S3 and S4. So, if there are solving a minimization problem; then there would have taken S4 as a better solution because it has a lower objective function value. Now, since there are working with the fitness value which is inversely related, there are supposed to take a solution that has maximum fitness. So, given two solutions if need to perform a greedy selection, there will have to select a solution which has higher fitness.

In the other algorithm, it was lower fitness because objective function value directly corresponded to the fitness function value whereas, in ABC, the objective function value and the fitness function value are inversely related. So, if there are solving a minimization problem then there will have to select a solution that is having a higher fitness.

Greedy Selection to update the solution,

Table 3.3.3: Formula Initials (3.2)

$X = Xnew$	if $fitnew > fit$
$f = fnew$	if $fitnew > fit$

Table 3.3.4: Formula Initials Description

(3.3)

X = Current Solution.
$Xnew$ = Newly generated Solution.
f = Objective function value of a solution.
$fnew$ = Objective function value of new solution.
fit = Fitness of a solution.
$fitnew$ = Fitness of new solution.

So, if the fitness function of the new solution is greater than the fitness of a particular solution then there will replace. So, X is equal to X new, f is equal to f new. So, this is the selection criteria that the fitness of the new solution has to be better than the fitness of the solution which is either undergoing the employed bee phase or the onlooker bee phase.

Table 3.3.5: Formula Condition (3.4)

X and f remains the same if $fitnew < fit$

If this condition is not satisfied like if the fitness of the new solution is not better than the fitness of the existing solution that is if the fitness of new solution is less than the fitness of the old solution then, there retain the same values for the decision variable and the objective function value.

So, this is a major difference between many of the algorithms and the ABC algorithm.

3.4 Employed Bee Phase

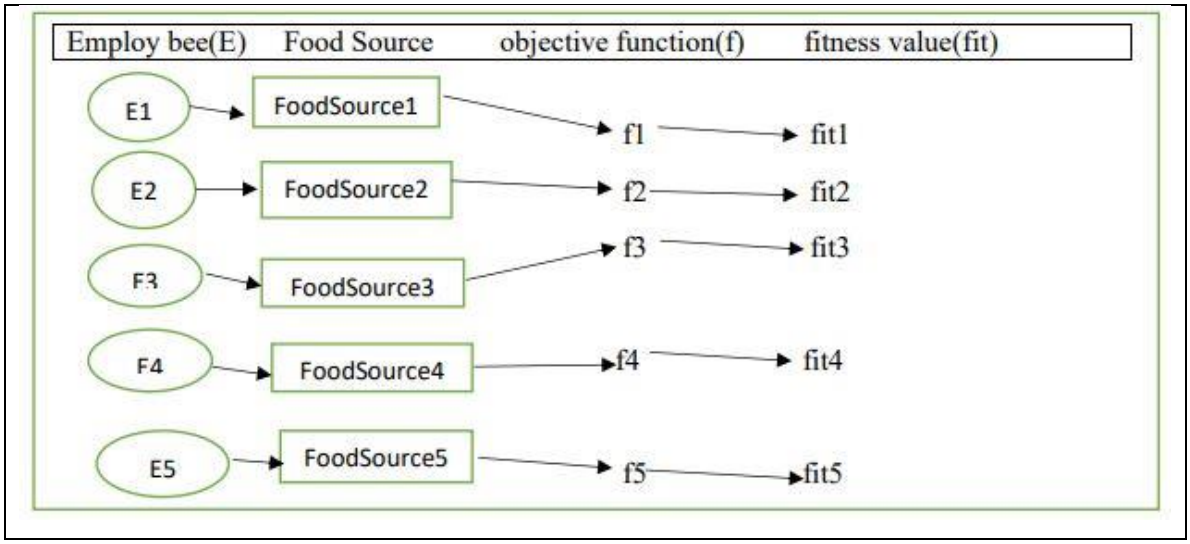

Figure 3.4.1: Employed Bee

There have 5 Food Source; FoodSource1, FoodSource2, FoodSource3, FoodSource4, FoodSource5 These are decision variables or population numbers this these are the decision variables that there have. So, these are the solution vectors which there have been calling population members. So, corresponding to each of these solutions there will have an objective function value. So, objective function value, there indicating by f1, f2, f3, f4, f5. these are the objective function values. Based on the objective function value, there also a calculation of the fitness value based on the formula. So, in the ABC algorithm, there need to fix something called swarm size. So, the number of food sources, the number of employed bees, and the number of onlooker bees is S by 2. So, if there fix

the swarm size as 10, then the number of food sources is 5. The number of employed bees is 5 and the number of onlooker bees is 5. So, in employed bee phase what will happen is this bee is exploiting this particular food source. The first bee is exploiting the first food source, the second bee is exploiting the second food source, the third bee is exploiting the third food source, the fourth bee is exploiting the fourth food source and the fifth bee is exploiting the fifth food source right. This is happening in the employed bee phase that there have 5 food sources, there have 5 bees. So, each bee is going to exploit a particular food source.

So, every bee is going to exploit a particular food source, that is one thing that needs to keep in mind for the employed bee phase.

3.5 Onlooker Bee Phase

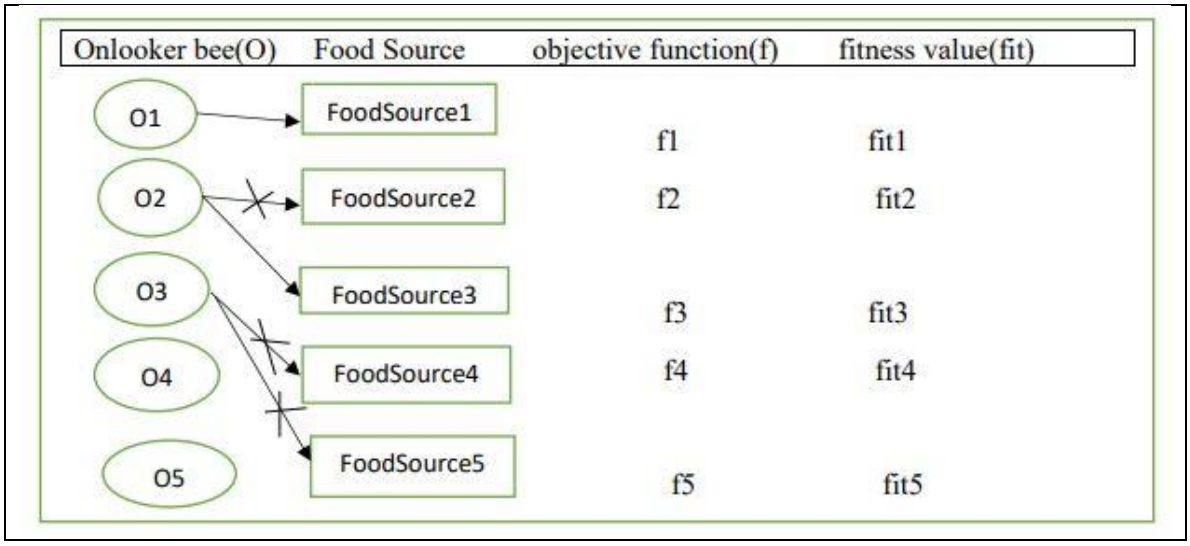

Figure 3.5.1: Onlooker Bee

In the onlooker bee phase; again there have the same 5 food source; FoodSource1, FoodSource2, FoodSource3, FoodSource4, and FoodSource5.So, these are food source. So, the food source in the employed bee phase and the onlooker bee phase may or may not be the same.

So, as an example, there will realize that why to say that the food sources may be the same or not. So, each food source that corresponds to the solution also has the objective function value denoted by f1 to f5 and there also has the fitness value; fitness1, fitness2, fitness3, fitness4, and fitness5. And there also have 5 onlooker bee, so let me call it as O1, O2, O3, O4, and O5.

So, remember in employed bee phase every bee did exploit the food source. In onlooker bee phase that is not necessary. So, first, they will start with onlooker bee1. So, onlooker bee based on a particular condition may or may not exploit food source. So, onlooker bee will exploit food source1 based on some condition, if the condition is met it will exploit. If the condition is not met it will not exploit food source1, but it will exploit foodsource2.

So, let us begin with let us assume that this condition is satisfied. So, onlooker bee1 is exploiting food source1, for the second onlooker bee; let us assume that this whatever that condition is not satisfied. So, since if this condition is not satisfied onlooker bee2 will exploit the food source3. Remember this did not happen in the employed bee phase. In the employed bee phase; each employed bee was exploiting a particular food source. There was a 1 to 1 correspondence. Here there is not that correspondence. So, if this condition happens to be true then this will exploit it otherwise this will not exploit it. Let us say O3is not meeting the condition of food source4 as well as it is not meeting the condition of food source 5. So, now if there see they have exploited all the 5 food sources, 3 has not completed the onlooker bee phase because it did not find a food source. So, in that case, we will again start from foodsource1, foodsource2, foodsource3, foodsource4, foodsource5. So, there will formally see it in a detailed way subsequently.

So, this is essentially the onlooker bee phase and the employed bee phase. So, in the ABC algorithm, there has a unique feature that keeps track of the number of failures.

3.6 Employed Bee Phase: Generation of New Solution

A solution is to generate a new solution. If there are generating a solution which is better, then the new solution is; taken inside the population because of the greedy selection, but if a solution is not able to generate a better solution, then there keep a track of the number of failures.

Now formally show here employed bee phase,

1. The number of employed bees is equal to food sources.
2. All solutions get an opportunity to generate a new solution in the employed bee phase.
3. A partner is randomly selected to generate a new solution.
4. Partner and the current solution should not be the same.
5. A new solution is generated by modifying a randomly selected variable.

Table 3.6.1: Generation Formula (3.5)

$$X^j new = X^j + \emptyset(X^j - X^j p)$$

Table 3.6.2: Generation Formula Description (3.6)

$X^j = j^{th}$ variable of current solution.
$X^j new = j^{th}$ variable of new solution.
$X^j p = j^{th}$ variable of p^{th} solution.
$\emptyset$ = Random number between -1 and 1.

This equation if there see X^j corresponds to the j^{th} decision variable which was randomly selected, j^{th} variable this was this j^{th} variable was randomly selected. X^j is the j^{th} variable of the current solution which is undergoing the employed bee phase and $X^j p$ is the j^{th} variable of the p^{th} solution where p is the partner. Again partner has to be randomly selected. In this relation phi is a random number between minus 1 and. So, now this variable may or may not be in bounds. So, if it is in bounds well and good, if it is not in bounds there will again employ the corner bounding strategy.

6. Bound the newly generated solution.

Table 3.6.3: Generation Formula Condition (3.7)

$X^j new = lb$	if, $X^j new < lb$
$X^j new = ub$	if, $X^j new > ub$

So, the corner bounding strategy is there are a lower bound, there are an upper bound if a solution is anywhere between the lower and upper bound, then there do not need to employ a bounding strategy. But, if it violates the lower bound there to bring it back to the lower bound and if a solution violates the upper bound there bring it back to the upper bound. So, this is the bounding strategy that there have been using in these techniques

3.7 Employed Bee Phase: Selection of New Solution

The procedure to generate a new solution either in the employed bee phase or in the onlooker bee phase is the same. There are select a random partner, and select a random variable and then modify just one variable to generate the new solution. The way to generate a solution in the scouting phase is different.

1. Evaluate the objective function and fitness of the newly generated solution.
2. Perform a greedy selection to update the current solution.
3.“trial” counter is used to track the number of failures encountered by each solution.

So, each solution has a trial associated.
If there have 5 solutions, there will have 5 values of trial. So, every time a solution encounters a failure its trial value is increased by 1. At any phase by looking at the trial value, there will be able to say that for how many times that particular solution did generate a new solution, but that new solution was not better. So, it gives a measure of how many times the solution has failed to generate a better solution.

4. Increase the "trial" of the current solution by one, if the new solution is inferior.
So, there will be having this trial vector. If the new solution is inferior there will increase the trial counter by 1 or if the new solution is better right. So, the new solution will come into the population, and since this is a new solution that has come into the population the trial is set to 0. This trial vector will be handled in the same way in the onlooker bee phase also.

5. Reset the "trial" to zero if a better solution is generated.
So, there will generate a solution, if the solution is good we will take it inside the population and reset the trial, if the newly generated solution is bad then there will update the counter by increasing its value by 1. So, it will keep track of the number of failures. The trial vector keeps a count of the total number of failures, irrespective of whether the failure happened in the employed bee phase or the onlooker bee phase.

3.8 Pseudocode of Employed Bee Phase

Table 3.8.1: Pseudocode Employed Bee (3.8)

Input: Objective function, lb, ub,Np,P,f,fit,trial
For ¡ = 1 to Np
Randomly select a partner(p) such that ¡ ≠p
Randomly select a variable j and modify j^{th} variable
Bound $X^j new$
Evaluate the objective function ($fnew$) and fitness ($fitnew$)
Accept $Xnew$, if $fitnew > (fit)$¡ and set (trial)¡ =0. Else increase trial by1
End

So, now let us look at the Pseudocode of the employed bee phase. So, in the employed bee phase we know that every solution has to undergo this employed bee phase. So, this for ¡ equal to 1 to Np. Np is the number of food sources or the number of employed bees or the number of onlooker bees which is S by 2 right; S is the swarm size. So, there have this external loop. So, there are going to repeat whatever is inside this for loop for Np times. So, there are going to do is randomly going to select a partner p such that ¡ is not equal to P. Because if ¡ is equal to p this equation this value will become 0. So, the effort that there are putting in to generate a new solution would be meaningless. So, there needs to ensure that the partner is not the same solution and then there need to randomly select a variable j.

So, it does not matter if this problem is having 4 decision variables or 100 decision variable or 10000 decision variable, there will be modifying only one variable to generate a new solution right. This equation which there are using is more of exploitation rather

than exploration. This is because there are modifying only one variable and very likely the new solution is in the vicinity of the current solution. So, once there modify this, there needs to apply this equation and generate a new value for the j^{th} variable and then need to check for the bounds. So, if it is within bounds; well and good, if it is not in the bounds will have to bound the solution. Once bound the solution it is ready to evaluate the objective function and also fitness because ABC algorithm, works based on fitness and not directly on the objective function value. So, now there have two solutions. So, the solution which there generated and the solution that is undergoing the employed bee phase right. So, there will do a greedy search. So, if the fitness of the new solution is greater than the fitness of the i^{th} solution, then we will accept the new solution. Since there are accepting the new solution, the trial counter has to be set to 0. If this condition is not satisfied, if the fitness of the new solution is less than the fitness of i^{th} solution, then there will increase the trial counter and discard the new solution. So, here is the old solution is retained, but they also have a track of how many times that solution has failed. So, this is the pseudocode for the employed bee phase, it is fairly simple to implement.

So, this is the generation phase wherein there are generating a new solution and this is the selection phase wherein there are selecting a better solution. Particle swarm optimization, genetic algorithm, teaching-learning based optimization, differential evolution. In all four of this objective function value is the same as the fitness function value.

3.9 Determination of Probability Value

1.Probability value of each solution to undergo onlooker phase is determined as,

Table 3.9.1: Probability Formula (3.9)

$prob_{\text{i}} = 0.9\left\{\frac{fit}{\max(fit)}\right\} + 0.1$	fit_{i}= Fitness of i^{th} solution Np = Number of food sources $prob_{\text{i}}$ = Probability of i^{th} solution

2. The probability values of all solutions are determined before the onlooker phase.

3. A solution with higher fitness value will have a higher probability.

4. Fitter solution may undergo an onlooker bee phase for more than once.

The onlooker bee phase, there is a condition that has to be met for a bee to exploit a particular food source right. So, for each food source, calculate something called as a probability. So, the probability is given by this formula. The probability of the i^{th}solution is 0.9 times the fitness of the i^{th} solution divided by the maximum fitness in the food source plus 0.1. So, the probability of all the solutions is to be determined before there undertake the onlooker bee phase. So, here can interpret that a solution that has a higher fitness value will have a higher probability. And as there will realize that a fitter solution has a greater probability of undergoing an onlooker bee phase for more than once.

3.10 Pseudocode of Onlooker Bee Phase

Table 3.10.1: Pseudocode Onlooker Bee (3.10)

Input: Objective function, lb, ub,Np,P,f,fit,trial
Set m=0 and n=1
While m<Np
Generate a random number r
if r< $prob_n$
Select a random partner (p) such that n ≠ p
Randomly select a variable j and modify j^{th} variable
Bound $X^j new$
Evaluate the objective function ($fnew$) and fitness ($fitnew$)
Accept $Xnew$, if $fnew > fit_n$ and set $trial_n$ =0. Else increase $trial_n$ by 1
m=m+1
end
n=n+1
Reset n=1 if the value of n is greater than Np
end

There discussed an onlooker bee phase using a pseudocode. So, initially, there set m is

equal to 0 and n is equal to 1. So, n is curve going to correspond to our food source, m is going to correspond to our onlooker bee. There will see why there are initializing m to 0 in a little while. So, there are supposed to generate Np new solutions in this example we are taking 5. So, we need to generate 5 new solutions. So, there use a while condition over here. Initially, there has not generated any new solution, so that is why there had initialized m to be 0. So, this condition will be satisfied till there do not get five solutions.

So, that way there will ensure that there are doing this loop with whatever is there within this while loop for Np number of times.

So, first what there needs to do is, there need to generate a random number r. If the random number r is less than the probability of n. So, initially, n is 1. So, first, there are checking the random number which there generated with the probability of Foodsource1. So, if that condition is satisfied, there will select a random partner there will select a random variable and modify the j^{th} variable using this equation which there have been using. And then since there has generated a new value for the decision, variable there needs to check whether it is in bounds. If it is in bounds it is well and good else, there will use the corner bounding strategy to bound it. After bounding it there need to evaluate the objective function since ABC works with fitness, there need to calculate the fitness of the new solution. Once again objective function in ABC is not the same as fitness, it is inversely related. So, there need to calculate fitness. If the solution which there generated is better that is if the fitness of the new solution is better than the fitness of the n^{th} solution. Then there need to accept the solution into the population and set trial to 0 else will increase the trial counter by 1. So, this way there have made a first attempt. So, there increase in the counter of m by 1, but if this condition was not satisfied, then there need to move on to the next food source. So, n is equal to n plus 1. So, the second time we are again generating a random number and checking the probability of this condition for the second food source. So, if analyze it might happen, as it happened in the example 4 m becomes 5 because there require five new solutions before m becomes 5, this n can go above 5. If this n goes above 5 then there do not have any food source. So, that is what there say, reset n equal to one of the value of Np is greater than Np. Since this can happen

we reset the value of n to 1. So, now, there have completed the employed bee phase as well as the onlooker bee phase.

So, if there think about it there started with a swarm size of let us say 10, then the number of food source is 5, the number of employed bees is 5 and the number of onlooker bees are 5. So, in the employed bee phase we generated 5 new solutions. In the onlooker bee phase, we generated 5 new solutions again as discussed previously, this is not a fitness function, this is the objective function. Because, ABC has a different meaning for objective function and fitness function whereas, in all other algorithms for unconstrained optimization problem objective function was the same as a fitness function. Again if there compare this with other metal heuristic techniques there have a generation phase over here and a selection phase over here.

3.11 Limit: User-Specified Parameter

1. Limit is a user-specified integer value.
2. Every solution is associated with an individual trial counter.
3. If the value of the trial is greater than the limit, the solution can potentially enter the scouting phase.
4. The trial counter of abandoned solution is reset to zero.
5. The value of limit can be set as limit =Np ×D where D is the dimension of the problem

So, like other algorithms, there needs to specify the number of cycles or the termination criteria for the ABC algorithm and there also needs to specify the swarm size. So, if there know the swarm size there can calculate a number of food sources S by 2, the number of food sources is equal to the number of onlooker bees and the number of onlooker bees is the same as the number of employed bee phase. In addition to these two parameters there also need to specify a parameter called as limit. So, it is an integer value. So, as there have seen that every solution is associated with an individual trial counter. So, for every solution there had a trial counter that is not set by the user, but that is just keeping a track of how many times a solution is failing to generate a better solution. So, this limit is an integer value. So, if the value of trial for a particular solution is greater than the limit

which has been set by the user, then the solution can potentially enter this scout phase.

So, not all the solutions will undergo the scouting phase. Only those solutions which have failed more than the specified number of times let us assume that there has set a limit value of 5. So, in that case, the solutions which have failed more than 5 times can potentially enter the scout phase not all of them will be entering, but only one of them would be entering. So, there look at it in a little bit later. Usually, the value of the limit is given by Np into D, where D is the dimension of the problem, and Np is swarmed size by 2 or the number of food sources. So, this is the most commonly used value for limit further research has been done on how to optimally set this limit if there are interested there can look into this, there will not go into detail of that. But, usually, the limit value is said to be Np into the number of decision variables.

Now, that there has the limit parameter, there can look into the scouting phase.

3.12 Scout phase

1. Solutions with trial greater than the limit are the candidates to be discarded.
2. One solution with its trial greater than the limit is replaced with a new random solution.
3.trial counter of the newly included solution is reset to zero.
4. In one iteration, scout phase,
4.1. This occurs only when the trial counter of at least one solution is greater than the limit.
4.2. Performed only on one solution with trial counter greater than limit
4.3. It can eliminate the best solution from the population due to the limit.
4.4. Memorize the best solution before performing the scouting phase.

So, one solution with its trial greater than the limit is replaced with a new random solution. So, there may have many solutions, but only one of them is replaced. So, that solution is discarded and a newly generated random solution is included in the population. Since a new solution is generated the trial counter for the new solution will be reset to 0. So, in every iteration or cycle, the scout phase may be encountered or may not be encountered. So, it occurs only when the trial counter of at least one solution is greater than the limit. So, if there, limit is 10 and there have 5 solutions whose trial counter is 2,

8, 7, 9, 3. So, it indicates that the first solution failed for 2 times to generate a new solution. Similarly, the other values can be interpreted. In this case, the scouting phase will not be encountered because all of these are less than 10. The scout phase may or may not be encountered in an iteration that depends upon the trial vector and the limit value. So, this scout phase is performed on only one solution. Let us say instead of this there had trial vector as this. So, now, when they're complete the onlooker bee phase, two solutions are exceeding the limit. So, even then only one of them would be replaced. So, that is why it is performed only on one solution with a trial count greater than the limit. What can happen in the scouting phase that the best solution can have a very high limit right? Let us say there encountered the best solution and let us say that the best solution is not able to generate better solutions either in the employed bee phase or in the onlooker bee phase. It is continuously encountering failures right and the number of failures is higher than the limit. So, it can potentially get eliminated. The best solution can potentially get eliminated from the population. So, that is why before there enter the scout phase there will memorize the best solution. So, even if it enters the scouting phase and it is discarded there will not lose that solution. Because, we have stored that best solution in a particular variable.

3.13 Pseudocode of Scout Bee Phase

Table 3.13.1: Pseudocode Scout Bee (3.11)

Input: Objective function, lb, ub,P,trial, limit 1.Identify the food source(k) whose trial greater than limit. 2.Replace X_k from P as, $X_k = lb + (ub - lb)r$ 3.Evaluate objective function (f_k) and assign fitness (fit_k)

Now, let us look at the pseudo-code of the scout bee phase right. So, there needs to identify the food source whose trial is greater than the limit right. there need to replace that the entire solution. To once there replace the entire solution, there need to evaluates its objective function and assign the appropriate fitness function, remember there is no greedy selection over here. Because that solution that has been selected to be discarded from the population has not been able to generate new solutions. So, the new solution

which there are generating over here may be worse than the solution that is being discarded. So, there are not employing a greedy selection strategy over here. So, that is how the scout bee phase is different from the onlooker bee phase and employed bee phase. One is that there change all the decision variable values whereas, in employed bee phase and onlooker bee phase; there were changing only one decision variable value. And secondly, both in employed bee phase and onlooker bee phase there employed a greedy selection strategy whereas, in scout bee phase there will directly take the newly generated solution into the population there will not employ a greedy selection strategy.

3.14 Selection of solution to perform Scout Bee Phase

Table 3.14.1: Scout Bee Performance (3.12)

Case 1		Case 2		Case 3	
F1 (trial=4)	(trial < 5)	F1 (trial=6)	(trial > 5)	F1 (trial=2)	(trial < 5)
F2(trial=0)	(trial < 5)	F2(trial=0)	(trial < 5)	F2(trial=8)	(trial >5)
F3(trial=2)	(trial < 5)	F3(trial=7)	(trial > 5)	F3(trial=8)	(trial > 5)
F4(trial=6)	(trial > 5)	F4(trial=2)	(trial < 5)	F4(trial=8)	(trial > 5)
F5(trial=5)	(trial = 5)	F5(trial=6)	(trial > 5)	F5(trial=0)	(trial < 5)

There look at three cases that can be encountered in the scouting phase. There consider these are the 5 solutions F 1, F 2, F 3, F 4, F 5 and these are their respective failures. So, solution 4 has encountered 6 failures so far and let there, limit value be 5. So, in this case, these four solutions, solution F 1, F 2, F 3, and F 5 have a trial value which is less than or equal to 5. So, they will not under goes a scout bee phase. So, the only solution which can undergo the scout bee phase is this one because it has a trial value greater than 5.

So, in case 2, if there see there are three solutions
F 1, F 3, and F 5 whose trial is greater than 5. So, one of these three solutions will undergo the scouting phase, the one that I will undergo is F 3, because it has the maximum number of failures.

So, in case 1 there was only one solution. So, there was no problem. In case 2, there were multiple solutions that violated this limit parameter right. So, in that case, we will select the one which has failed the maximum number of times. So, this is also violating this condition, this is also violating the condition, but these two food sources will not undergo the scout phase because in the scouting phase only one solution has to be discarded.

So, in case three, these three solutions have identical trial values 8 8 8. So, all of them are greater than 5 which is our trial limit. So, in this case, there need to randomly select one of these solutions. So, all of these have a maximum number of trails. So, there will select randomly one solution because they have the same number of maximum trials which is also greater than 5. So, this is all of these are greater than 5. So, anyone of this can be selected. These are the three cases in which there will have to decide which particular solution undergoes the scouting phase.

3.15 Pseudocode of ABC

Table 3.15.1: Pseudocode ABC (3.13)

```
1.Input: Objective function, lb, ub,Np,T and limit
2.Initialize a random population (p)
3.Evaluate objective function (f) and fitness (fit)
4.Set the trial counter of all food sources equal to zero
   for t = 1 to T
        Perform Employed Bee Phase of all food sources
        Determine the probability of each food source
        Perform Onlooker Bee Phase to generate Np food sources
        Memorize the best food source
        if trial of any food source is greater than limit
          Perform Scout Bee Phase of exhausted food source
        end
end
```

So, the entire pseudocode of ABC can be given as this thing that there needs to first initialize a random population within the bounds of the decision variable. So, the bounds should be known and there need to have a fitness function. So, that there can calculate the objective function right fitness function as in like objective function is required and then there will use the relationship between the objective function and the fitness function to calculate the fitness of every solution. And then there needs to set the trial counter of all food sources equal to 0. So, this trial counter will keep track of the number of failures. So, if the number of food sources is 5, the trial is a vector of 5 elements. ABC is an iterative technique. So, there will have to perform the steps of ABC multiple times, it is either known as the cycles or there can call it as iterations. So, there will have to perform it for t iterations will have to first perform the employed bee phase on all the food sources and then there will have to determine the probability of each food source and then there perform onlooker bee phase to generate Np food source. In employed bee phase all the food sources are going to definitely be used to generate a new solution that is not the case in the onlooker bee phase. But still there need to generate Np food sources. Once there are done with that there need to memorize the best food source. This is an important step because the best food source can be lost in the scouting phase. So, there

needs to memorize the best food source and there needs to check if the trial of any food source is greater than the limit. There need to first determine if the scout phase is to be performed using the trial vector and the limit value which has been set by the user. If the condition is, meet then there perform the scout bee phase of the exhausted food source. So, this completes the pseudocode of ABC.

3.16 Working of ABC: Sphere Function

Table 3.16.1: Sphere Function (3.14)

$f(x) = \sum_{i=1}^{4} (Xi)\text{^}2$	$[0 \leq (Xi) \leq 10]$

There have been taking for all the other metaheuristic techniques, the spear function. So there will take the spear function with four decision variables. So, the spear function is the summation of the square of all the decision variables. So, the first step is to fix this swarm size the number of cycles, and the parameter limit. So, the number of swarm size will tell the number of food sources, the number of employed bees and the number of onlooker bees. The number of cycles will tell us when to complete our procedure, and the parameter limit helps in determining as to when the scout phase is to be implemented.

CHAPTER 4

RESULT AND DISCUSSION

In the simulation studies, Artificial Bee Colony (ABC) Algorithm was applied for finding the global minimum of the well-known functions. One of the functions is Sphere function that is continuous, convex and unimodal function.

In this section, the experimental results of the proposed method in comparison with the best result at present hive with different iteration that are done by bees. There present are presented. the best-solution, best-objective, hive position. The sample itself also calculated there have all Performance percentages. The number in the Np of each sample means the number of food source used in the sample. Each sample was executed 10 times with the same different number of food source used. So, the size of food source (10,20,30,40,50) the number of iteration (1,2,3,4,5,6,7,8,9,10) limit (5) and lower and upper bounds are [0 0] [5 5] were used. However, the maximum best performances are counted and calculated the coverage performance percentages.

The running environment consisted of a desktop PC with Windows 10, 2.50 GHz Core 2.71 Duo CPU, and 8.00 GB RAM. The ABC strategy was coded and implemented in MATLAB (R2019a).

This process was continued individually 10 time for all food sources. The data in Table 1 summarize the attributes of the all the best solutions in this study. The underline and bold numbers are indicated that the best obtained result for the configuration of interests.

Table 4.1: Result

Np	T(iteration)	Best-Solution	Best-Objective	Position	F
10	1	1.5828; 0.4462	0.1991	1	**0.1991** 10.2700 5.8451 13.7925 28.7779 21.3761 30.5888 27.6248 22.6696 4.4081
10	2	0.3874 ; 1.8182	3.3060	5	13.7905 9.0634 10.1560 21.4879 **3.3060** 14.5213 18.6646 17.5455 26.4705 16.7805
10	3	0.4002; 0.1033	0.0107	8	22.6241 27.0539 22.3116 16.6552 25.6975 15.3461 11.4407 **0.0107** 12.2936 16.0522
10	4	0.7016; 0.3400	0.1156	11	27.1849 24.2269 3.1739 18.8059 23.8285 0.9942 15.4334 24.3266 0.1180 15.5059
10	5	0.7877;	1.3837e-05	11	11.5832

		0.0037			19.4552 16.6678 32.9139 13.6261 6.8033 14.8874 24.2671 16.1322 6.3756
10	6	0.9953; 0.7939	0.9770	9	9.7798 27.9695 8.1939 15.4519 18.6838 23.9032 17.5006 30.2083 **0.9770** 12.2417
10	7	0.3709; 1.2414	0.3506	11	13.2162 18.4772 12.6469 7.7075 12.4326 9.3404 17.0093 20.2846 3.2001 12.4619
10	8	0.6977; 2.4370	0.4906	9	28.5888 20.7831 3.8771 33.7224 2.4733 21.8756 14.9409 3.7950 **0.4906** 29.9292
10	9	0.2358 ; 1.0776	0.0556	11	17.7696 16.5115 3.9347 7.2587 9.5225 22.9348

					16.3061 7.1364 12.8379 17.0709
10	10	0.7264; 0.3551	0.1261	11	22.9821 12.3197 21.4586 8.9264 15.2114 35.9174 9.7236 17.0117 3.0972 28.3152
20	1	0.7798; 0.2700	0.0729	17	20.0315 11.0213 24.4439 17.5943 23.2963 4.0312 18.8882 20.4009 2.4845 15.8049 1.9764 20.2062 22.3831 1.7501 8.0322 15.1875 **0.0729** 16.6618 20.3905 2.3332
20	2	1.1900; 0.5932	.3519	21	9.6283 12.0782 2.6101 8.4158 3.7376 32.7878 3.0940 36.1965 2.0457 22.8323

					14.8276 10.9117 6.7289 1.7852 27.2869 8.4195 8.5438 5.8738 13.3832 14.283
20	3	1.3971; 0.4897	0.4419	21	5.4330 12.5376 4.7911 21.5533 0.8211 0.8262 29.1451 13.1861 18.5088 6.9719 14.7978 11.9147 8.8774 21.7177 25.5699 7.9756 3.5936 8.7484 23.6994 15.4372
20	4	0.7995;2.2112	.9854	9	17.7089 9.2712 12.3728 15.5902 18.5320 6.2133 15.2617 27.1963 **0.9854** 7.0236 11.9134 7.1955 13.4759 33.3661 1.2847

					1.8467 18.9468 9.9485 14.8383 16.843
20	5	0.1097; 0.8970	0.0120	21	19.0095 4.6683 9.3957 33.0406 1.8504 5.3527 7.5697 13.6629 9.8155 7.0245 8.9287 14.1563 14.8082 26.0669 9.3946 0.3595 11.2724 31.7641 4.6269 0.335
20	6	0.0874; 1.2922	.3001	21	12.1581 15.6205 9.3698 4.7275 12.6804 0.6320 6.8422 22.9812 2.3116 18.7926 21.5148 27.6966 10.1923 11.2889 20.9973 20.0876 19.6545 22.8565 13.7409 19.968

20	7	0.3101; 1.1064	0.0962	21	11.1991 14.1832 0.1646 28.3961 14.7925 19.8595 28.6191 37.7306 11.6703 15.7001 14.3413 16.0518 4.3966 8.8540 18.4557 15.4960 7.8038 6.8529 3.0193 10.179
20	8	0.1356 ;1.0203	.0184	21	24.7838 4.0307 12.1250 6.5407 13.4273 1.3153 0.8851 10.1490 3.2889 19.4234 7.9932 18.2721 23.9621 13.0609 3.2783 15.8791 18.2622 21.0920 11.2225 16.0873
20	9	0.2520 ;0.6267	.0635	21	3.4669 23.3593 13.8302 3.0798 11.8060

					19.5424 11.9219 1.1468 16.2026 2.8308 15.8309 23.4363 19.8552 6.9627 2.1354 12.427 8.9833 18.1456 0.2613 7.114
20	10	1.3206; 0.3322	.1104	21	10.6248 16.0167 16.5266 12.2091 0.6971 22.6643 11.7088 1.1285 17.4356 15.1540 9.9297 13.8762 13.8748 13.3256 9.2100 17.4572 6.6725 34.4052 3.7890 17.236
30	1	0.6272 ;0.0217	0.0672	24	10.0296 14.9571 27.1408 13.0789 6.9427 19.9914 22.5719 17.1388 16.9001 20.9598

					19.7290 6.2646 15.0086 7.1946 24.0791 39.7524 6.9203 14.3340 15.8979 31.2412 25.7452 4.3865 9.1398 **0.0672** 17.1436 25.3755 6.4236 1.7699 18.3270 2.1795
30	2	0.8462 ;0.0090	0.0754	16	9.8768 12.4064 33.0426 5.6206 4.6432 9.3446 19.2805 1.0263 24.4897 2.0687 23.6867 0.4592 23.5300 25.4526 4.1650 **0.0754** 29.1421 16.5001 0.3499 8.3686 24.6886 18.6828 11.2901 2.8307 11.2097

					22.4504 24.3985 26.2228 3.9332 3.9178
30	3	0.3187 ;0.5225	0.1016	31	9.3672 10.9851 4.0872 2.0718 0.3132 6.0485 5.8840 33.5637 14.7202 10.6224 6.6125 9.8977 12.5877 10.0577 10.0919 12.8123 23.3458 8.7698 33.6196 21.3840 10.1581 12.9694 28.1648 6.9066 18.2307 4.6501 17.1847 7.8220 22.7685 14.9063
30	4	1.1538;0.0723	0.0052	31	28.7443 27.3512 3.0598 15.7055 12.5042 20.9575 17.0731 18.1656 6.5287 28.6031

					8.8584 14.1467 16.1662 22.1713 4.2824 0.3139 12.0624 6.2722 3.7175 4.6938 12.3881 28.4955 11.2429 6.1125 22.3980 22.3247 4.3062 27.2574 0.8100 21.6700
30	5	0.0499 ;0.4389	0.0025	31	13.0554 4.6856 11.6709 14.7259 20.6433 13.0581 12.6840 11.0850 3.2704 40.9399 1.0688 15.5250 4.9856 13.1917 2.1985 23.4300 18.2440 9.0925 31.6426 3.8986 9.3880 8.3030 21.0242 22.3796 25.4344

					5.1175 1.4107 13.7580 11.1395 18.4817
30	6	2.4961 ;0.2924	0.1769	8	28.4211 15.4658 12.4355 9.5588 4.6477 16.8895 11.1338 **0.1769** 11.4800 7.4486 8.9716 8.6554 13.8918 5.7543 7.9804 22.6950 4.0890 8.1176 13.9467 4.2924 13.5951 9.7040 3.0730 12.6615 2.8868 13.4089 24.9049 0.3304 7.1769 17.2403
30	7	2.3738 ;0.1338	0.5654	31	19.5803 24.7831 6.6321 1.4644 5.1125 18.1517 2.3836 18.4104 20.4541 15.9182

					12.0442 3.9259 4.1948 15.6375 33.2136 12.1457 3.8339 8.8268 13.3066 13.6468 8.8307 5.2305 17.1700 1.8292 35.7486 21.2083 2.5519 27.0308 18.9171 12.6818
30	8	1.5238 ;0.0130	1.6915e-04	31	4.3185 7.8687 25.6761 17.1326 14.2201 1.0357 9.5596 17.2886 10.4868 15.2535 4.8398 3.1293 5.1911 22.5135 20.6678 13.7056 13.2628 4.9573 9.0424 7.4132 20.9413 1.9918 18.6784 18.0301 0.4667

					0.4834 24.2947 2.6094 15.5939 0.2828
30	9	0.5786 ;0.2744	0.0753	31	18.7718 13.8495 14.5957 11.8505 29.3746 20.1218 19.5966 17.5625 3.4988 32.1838 7.3569 12.3182 12.9653 1.6892 13.1000 6.1182 20.1514 5.3266 26.8110 12.8575 28.9004 18.1543 0.5541 21.4180 5.6497 8.2875 19.7120 18.4629 12.3301 10.9472
30	10	1.3465 ;0.0020	3.9423e-06	31	4.3526 25.5084 8.3502 23.1530 12.0041 14.6763 17.9188 6.1103 1.5214 10.4351

					10.6075 9.2990 10.9045 5.6757 2.1979 0.1550 11.8606 0.7646 1.5222 6.4557 0.0202 12.1753 10.2711 6.0522 5.5978 5.1116 22.7237 40.4889 4.2896 11.8082
40	1	0.2732 ;0.2291	0.0746	15	10.5280 34.5368 13.2024 4.2973 15.6144 21.8228 5.7064 17.1994 10.4328 13.0562 11.1976 27.2439 6.5862 11.6011 **0.0746** 16.1672 21.4032 6.4504 2.7014 9.8334 7.5886 0.2132 7.8596 15.0122 21.9752

					26.0257 5.6967 22.6119 12.9695 15.8887 10.5250 7.5474 30.6175 7.2400 18.9432 5.9750 11.8670 18.2322 17.1241 10.728
40	2	1.5492 ;0.1129	0.6386	7	0.8693 30.2554 7.8083 29.0613 12.5764 15.5692 **0.6386** 16.9537 3.9330 11.7619 10.6374 2.7384 14.3301 12.4170 1.3957 8.7232 5.5336 15.1216 12.9055 28.1106 8.8794 18.2112 9.5238 28.9605 5.9996 33.3744 2.2663 10.3839 14.3132

					4.0220 27.7773 13.5513 17.2177 18.5733 1.5184 11.4555 6.1198 16.4878 10.0032 5.8188
40	3	0.0673 ;0.6025	0.0045	41	1.9314 18.8404 22.6121 29.3921 17.4240 6.5690 5.3798 12.3319 8.3645 7.8015 19.2907 1.0758 5.4176 23.3127 11.4188 19.3799 1.8758 25.1840 8.7357 1.9048 20.4797 4.2982 9.5593 20.1950 3.4993 9.5908 22.9192 38.0502 39.0659 6.9170 0.1941 5.7899 2.7953 21.0480

					3.7139 5.2702 18.0234 6.7816 28.8041 3.2646
40	4	1.7931 ;0.2849	.1901	41	3.5259 8.5690 21.7308 9.8983 12.3241 6.0667 11.8273 25.8051 21.8744 16.8788 23.5449 2.4401 3.7113 10.4239 5.5563 11.2911 20.8419 13.1241 13.2558 10.4119 20.7175 8.8387 5.5905 14.2862 5.9909 3.0114 6.7342 8.1191 2.1899 4.3143 11.3602 4.8088 0.2822 10.8116 6.9860 6.9853 16.2172 1.6633 8.6955

					15.142
40	5	0.3835 ;0.0591	0.0035	41	0.7316 8.0392 1.0427 21.3774 7.2953 26.4066 18.4281 12.5017 7.4564 2.9822 7.9553 9.5252 21.8095 16.5427 1.3189 9.0922 1.4604 17.3026 10.6039 14.2946 3.7511 10.1184 1.9536 14.9588 10.4247 7.6443 13.8531 22.0305 5.3085 10.1635 18.7300 18.0338 11.6535 4.5610 14.3029 20.3511 22.0671 5.3148 13.1314 2.884
40	6	0.0035 0.1082	.0117	29	22.9653 16.7235

					25.7095 3.4030 17.2207 22.9957 18.4884 9.8517 22.3630 8.9144 12.6886 14.9123 4.7844 6.3896 3.6461 17.4741 19.2038 3.1291 3.4850 2.7361 0.7863 10.9242 6.0990 4.3673 7.2359 11.2180 0.8295 13.5390 **0.0117** 8.9705 28.5114 21.0149 25.6399 9.9034 1.2646 5.2774 11.7377 8.5559 22.5302 17.101
40	7	1.8748 ;0.0592	.0289	24	9.3646 22.5767 17.4288 16.8347 9.3646 22.5767 17.4288

					16.8347 3.6728 8.0879 6.9945 7.0867 23.4715 9.5931 12.0250 21.5389 20.3358 13.7920 9.5539 9.6480 10.3975 28.2699 4.7540 25.4299 1.0371 4.9109 3.4744 **0.0289** 11.3770 21.7545 8.0573 23.8639 11.3248 17.5835 6.0940 8.5138 4.4875 9.7561 20.8519 11.3227 15.8012 6.0138 14.1554 16.027
40	8	0.3121 ;0.4730	0.3211	33	12.4323 10.9934 3.6219 22.1192 23.6410 5.8371 16.8901 34.0211

					13.9986 2.6273 13.6049 12.4323 10.9934 3.6219 22.1192 23.6410 5.8371 16.8901 34.0211 13.9986 2.6273 13.6049 16.1399 6.3396 8.2529 11.1249 0.3686 31.6559 11.9132 10.4366 18.4614 38.4162 7.8593 16.8436 32.9434 9.3668 2.3057 7.3459 2.1034 8.5711 7.2874 22.6812 12.6319 **0.3211** 10.9069 3.8904 17.4346 13.7047 7.2488 2.3586 1.879
40	9	0.1584 1.6018	.0274	41	14.0834

					8.4533
					9.7754
					10.5159
					4.7806
					3.0278
					1.0302
					26.1979
					19.0558
					5.4427
					20.8075
					22.6841
					3.8276
					14.5673
					8.1439
					7.0080
					12.2119
					14.0834
					8.4533
					9.7754
					10.5159
					4.7806
					3.0278
					1.0302
					26.1979
					19.0558
					5.4427
					20.8075
					22.6841
					3.8276
					14.5673
					8.1439
					7.0080
					12.2119
					12.8559
					6.2403
					1.9019
					7.0738
					18.6183
					13.0812
					3.2045
					11.2713
					24.5742
					14.8882
					6.5225
					2.8012

					14.8104 5.6818 0.3354 26.1943 11.8990 3.5742 15.0226 20.9852 3.2130 16.7218 5.329
40	10	1.7378 0.2274	.0517	41	4.0763 18.0086 2.1909 11.8015 18.9822 21.5296 2.3170 27.1563 13.0229 3.4560 14.7543 13.9124 1.1904 10.9044 16.1934 6.3271 0.4103 19.7308 12.3163 17.3913 13.0630 8.8599 5.9430 10.5856 5.8095 38.9018 3.8317 8.9498 6.4183 1.8801 4.8335 10.9121 12.0832 24.6325

					0.9222 13.8254 11.8103 1.5650 2.4696 7.2712
50	1	1.3771 ;0.4426	0.1959	38	1.6295 25.9189 2.2501 25.3270 22.8048 14.1042 31.7870 18.7021 5.9820 16.6152 11.5092 0.6774 5.3554 3.0560 10.8652 17.7213 10.1223 17.0716 0.6683 42.6088 16.6490 3.8708 18.1794 29.2080 8.6906 3.9235 19.3864 15.9268 7.6877 6.4868 14.1400 14.8081 12.7051 27.0709 31.8957 3.7313 8.1190 **0.1959** 15.1572

					3.9174 22.2549 26.2429 9.2165 1.9100 4.8784 22.6812 8.1065 26.8944 18.4627 19.9808
50	2	0.0576 ;0.0953	0.0033	47	5.3292 4.4476 8.4959 2.3357 29.0506 10.6259 26.3900 4.9497 0.0069 10.5136 21.5157 3.1533 23.2289 23.5335 5.6157 28.9483 25.1074 18.2833 13.0959 8.6913 13.2393 0.1121 9.7352 40.0635 35.9865 27.3571 15.4761 1.5510 24.1631 7.6610 26.7497 11.7760 13.3472 13.1624

					27.0150 7.7754 12.7344 21.9675 12.7655 0.1366 8.4523 2.8149 24.0081 15.3886 0.0578 12.9632 **0.0033** 17.2575 21.8509 15.4901
50	3	2.0430 ;0.3065	0.0939	39	24.5681 16.1994 18.0586 9.0549 7.9606 13.0231 28.9332 1.4760 10.4721 3.8799 9.5890 12.7334 24.3838 12.0258 12.8108 29.8301 17.3961 5.1555 3.5636 5.5584 2.6623 0.0963 2.7724 9.8162 7.2929 19.1163 11.0783 21.4748 5.4138

					7.7001 28.0414 11.5550 7.5225 30.1231 3.6309 38.5263 8.9369 12.2881 **0.0939** 4.9634 5.3594 3.2131 0.5560 5.4464 17.1460 15.1374 7.6165 3.7664 11.3967 4.3818
50	4	0.0513 ;0.0231	5.3501e-04	51	13.4425 13.6281 7.3032 24.4369 14.8026 0.8023 4.2552 30.3037 14.4424 10.1113 21.3622 18.7533 20.0917 2.3331 12.1461 0.1997 26.6853 4.6002 18.2617 12.5989 9.8789 22.4387 28.8019 5.2246

					4.8846 7.1010 24.2688 11.0647 4.5761 8.7632 14.2712 6.1763 4.7353 17.2013 5.6073 5.1350 6.1707 13.4858 13.1065 14.7496 9.1968 29.0424 5.7275 31.0909 27.8148 9.2894 2.3453 17.5600 22.7894 11.6195
50	5	1.1452 ;0.0640	0.0041	29	8.6195 11.3694 2.4059 25.6567 0.0398 10.4948 0.6325 0.9536 11.0400 3.8125 8.6291 26.7098 6.3427 3.0048 14.1899 32.3784 17.8046 7.9970 6.8941

					5.1969 9.9873 13.2717 12.4219 23.9124 21.5164 6.2901 21.3925 11.6075 **0.0041** 19.3691 4.8533 27.0405 8.4859 16.2427 12.0822 24.8004 6.2831 12.0453 5.7374 4.6672 11.3652 0.0822 16.0533 6.8922 20.8427 6.7602 6.6662 14.4764 8.3265 27.7905
50	6	1.0502 ;0.0671	0.0045	51	18.0569 2.9634 11.2514 6.2899 3.0115 1.3083 8.5672 2.5622 5.9549 21.8980 15.2532 14.4671 34.6084 9.3947

					8.4633 7.6665 2.7452 0.1670 11.0890 3.4028 0.7482 5.8548 1.2262 12.3884 6.0781 10.6185 26.9119 16.9764 2.4605 0.1365 23.4560 9.1692 4.8534 7.1266 20.3049 3.9535 25.6899 14.3345 6.2464 9.9145 8.2991 6.5821 10.5941 11.0219 8.5425 9.8249 10.7347 0.5143 16.3457 14.3724
50	7	0.0111 ;1.5306	1.2304e-04	51	6.1621 12.7477 12.3535 0.6999 23.3149 7.9564 6.4541 8.7085 2.9192

					2.7395 11.5489 4.2450 10.4099 4.5927 1.3609 7.2031 10.1682 22.0571 5.8463 8.6913 1.6419 1.4632 12.4875 3.2695 16.6166 8.7103 7.6640 9.8697 21.2151 12.6102 13.7758 11.3646 5.8194 6.7720 15.6474 14.3440 20.1482 12.3378 10.6026 9.4038 6.8041 18.1531 0.6456 1.0924 25.9260 8.0331 0.7087 5.7952 5.5408 0.0120
50	8	0.2534 ;1.0910	0.0642	51	15.5956 9.5714 11.5170 8.9453

					3.2210
					9.8758
					15.2125
					7.1848
					5.3118
					5.3063
					15.0824
					9.1319
					3.5859
					12.5156
					3.0705
					4.5999
					9.6381
					1.9236
					0.3644
					15.3725
					10.8389
					15.0408
					0.9737
					3.3270
					4.3139
					2.8181
					0.1925
					7.7251
					8.1331
					24.0804
					12.0558
					11.8091
					9.1807
					19.9224
					0.9015
					0.2724
					23.1902
					6.5977
					1.0400
					22.1917
					10.3314
					1.9031
					23.2084
					20.6433
					5.3491
					4.9290
					0.9303
					4.1485
					25.4821

					8.2467
50	9	0.8405 ;0.1107	0.0123	43	7.6832 5.0198 11.7847 2.1199 1.1397 8.9780 6.1378 15.7995 10.1449 0.4804 5.7188 17.5539 28.3361 24.1880 6.7174 3.3424 12.6832 15.6288 12.4488 10.3330 0.8101 2.8955 0.6534 3.3841 0.0555 6.6938 11.8837 21.6398 9.5814 10.6740 0.2321 2.4044 12.5247 3.8585 19.1759 4.1847 9.8667 13.5495 5.0203 14.0837 14.9106 0.8510 **0.0123** 3.8405

					25.5372 2.8071 14.0429 1.1768 0.8989 8.1519
50	10	0.1166 ;1.0984	0.0136	51	20.1778 7.8852 21.4576 5.2977 23.8908 13.7055 22.3798 17.8587 22.9979 25.7691 15.6239 16.4795 1.4448 32.1258 5.9569 16.7784 16.2350 10.5033 12.8712 22.6719 13.0467 3.9575 17.8781 16.5397 13.7007 7.0386 5.9728 13.4995 17.2228 17.6649 21.2094 23.4651 20.4631 46.4008 24.2902 5.1567 1.4623 12.3160 19.8398

					4.3821 0.2949 20.0725 7.6307 3.2954 30.4378 19.5347 7.5377 12.7879 5.3577 11.1417

In this Result, we present and discuss the results generated by ABC algorithms. In our experiments, we compute the average percent above the optimal solution (average of 10 runs) for the algorithm on each food source. Next, we calculated performance coverages over the problems. In Table 2, we display five different food sources and record the total running time (in hours for all 50 runs on all 5 food sources). In Table 2, overall 5 performance coverage, we see that 10 number of food sources were the most coverage performance.

Table 4.2: Coverage

Np	Performance percentages
10	50%
20	10%
30	10%
40	12.5%
50	10%

However, this ABC strategy produced competitive and very close results to the optimal result for the rest of the configurations systems. This result clearly has shown that

food source(Np)=10, with (1-10) iteration there, find 5 valid best-objective.

Food source(Np)=20, with (1-10) iteration there, find 2 valid best-objective.

food source(Np)=30, with (1-10) iteration there, find 3 valid best-objective.

food source(Np)=40, with (1-10) iteration there, find 5 valid best-objective.
food source(Np)=50, with (1-10) iteration there, find 5 valid best-objective.

According to this result, we show that when the food source is 10 and the iteration also (1-10). Then the performance coverage is highest. It's 50%, and when the food sources are increase but the iteration limit is fixed then the performance decreases. So whenever the food sources are increasing then it needs to increase the parameter of T(iteration), then there best objective value will be an increase, moreover, the best objective values indicate that the best optimal result.

ABC algorithms are mainly used there for optimal the result, and when the bees also find the best optimal result in which the number of positions. Here when the food sources are 10 then the bees are finding there, the best result in the position of (T-1) (position-1), (T-2) (position-5), (T-3) (position-8), (T-6) (position-9), (T-8) (position-9), when the food sources are 20 then the bees are finding there, the best result in the position of (T-1) (position-17), (T-4) (position-9), when the food sources are 30 then the bees are finding there, the best result in the position of (T-1) (position-24), (T-2) (position-16), (T-6) (position-8), when the food sources are 40 then the bees are finding there, the best result in the position of (T-1) (position-15), (T-2) (position-7), (T-6) (position-29), (T-7) (position-24), (T-8) (position-33). when the food sources are 50 then the bees are find there, best result in the position of (T-1) (position-38), (T-2) (position-47), (T-3) (position-39), (T-5) (position-29), (T-9) (position-43).

So, based on this result we assume that, when we find the best solution for any food sources, then we have to select a current number of iteration, and we have to select the iteration numbers based on that, how many food sources are given there. Then we find the max number of best solutions from those food sources.

CHAPTER 5

CONCLUSION AND RECOMMENDATION

In Chapter 4, the experimental results of ABC strategy are discussed as the foundation of the expected goals and highlighted the contributions achieved from this research in the field of software test engineering. This chapter summarizes the important findings of the developed ABC. Finally, this chapter ends with the future directions of this research. This research has proposed ABC for pairwise test suite generation by relying on a hybrid artificial bee colony algorithm. Through the conducted experiments, ABC has shown a good performance by producing the optimal test suite size for some configuration systems.

5.1 Implication

An artificial bee colony is a new optimization technique that has shown to be competitive with other population-based stochastic algorithms, be that as it may, ABC and other stochastic calculations experience the ill effects of similar issues. The primary explanation is that the inquiry example of both utilized and spectator honey bees is acceptable at investigation yet poor at abuse. In order to balance the exploration and exploitation of ABC, this paper proposes a new ABC strategy. It has resulted that more iteration is made easy to mutation scheme is good at exploitation. Our approach differs from other ABC algorithms strategy. To verify the performance of our approach, a set of Sphere benchmark functions are used in the experiments. A comparison of the different iteration of different food sources with ABC demonstrates that our new search pattern can effectively accelerate the convergence speed and improve the accuracy of solutions.

Contrasted with other improved ABC calculations, our methodology is less complex and simpler to execute.

5.2 Limitation

In this paper, the Artificial Bee Colony algorithm is proposed in solving issues in the t-way testing field. The algorithm consists of 4 phases which are initialization, employed bees, onlooker bees scout bees, and. The algorithm will be implemented and the performance will be compared against other strategies using Sphere benchmarking experiments. In this paper, we have designed and implemented an artificial bee colony algorithm for the best solution with finding the position of best-objective based on iterations. The convergence speed of ABC is typically slower than PSO and DE. Besides, the ABC calculation effectively stalls out when taking care of complex multimodal issues.

Analyze, after (or during) testing, test results need to be analyzed. In this paper there focused that when the iteration is selected based on food sources then there have the more valuable best solution with the position, and the limitation of this paper is that according to the food source if the limit of Sphere function and the lower bound and upper bound are not selected then the value of the best solution cannot find a proper way so that finding the best position with the objective is find when all over function parameter used properly, that's completely a mathematical way of finding the optimal result of bees food sources with our ABC strategy. This can reduce the testing cost, compared to exact approaches, yet improve accuracy, From the reenactment results, it was presumed that the proposed calculation can escape a nearby least and can be effectively utilized for multivariable, multimodal work enhancement.

5.3 Future Plan

Conclusively, a survey has been carried out on the most-recent strategies implemented for t-way testing. This survey has analyzed and highlighted the strengths, weaknesses, and limitations of each of these strategies. Owing to this, a new t-way strategy, known as the ABC, has been suggested. This new technique uses a counterfeit honey bee province calculation as a center calculation for execution. The first outcome demonstrates that the ABC is capable of outperforming some existing strategies. As part of our plan to improve the design of the ABC in the future work.

Several issues remain, as the scopes for future studies such as the investigation of the control parameters' effect on the performance of the ABC algorithm and the convergence speed of the algorithm. We plan from the start to cover high cooperation boundaries. We plan from the start to cover high cooperation boundaries. Then, we intend to the artificial bee colony algorithm with other Optimization algorithms to develop its search capabilities in all aspects.

As part of our future research, we are looking to extend the ABC strategy for supporting high interaction strength to use in the future for Software Product Line (SPL) as well as to support the variable strength interaction. And also the algorithm will be applied to more complex test problems and this algorithm will be hybridized with some local search heuristics to find better results.

REFERENCES

Ahmed, B. S., Abdulsamad, T. S., & Potrus, M. Y. (2015). Achievement of minimized combinatorial test suite for configuration-aware software functional testing using the cuckoo search algorithm. *Information and Software Technology*, *66*, 13-29.

Akay, B., & Karaboga, D. (2012). A modified artificial bee colony algorithm for real-parameter optimization. *Information sciences*, *192*, 120-142.

Alazzawi, A. K., Homaid, A. A. B., Alomoush, A. A., & Alsewari, A. A. (2017). Artificial bee colony algorithm for pairwise test generation. *Journal of Telecommunication, Electronic and Computer Engineering (JTEC)*, *9*(1-2), 103-108.

Alazzawi, A. K., Rais, H. M., & Basri, S. (2018, August). Artificial bee colony algorithm for t-way test suite generation. In *2018 4th International Conference on Computer and Information Sciences (ICCOINS)* (pp. 1-6). IEEE.

Alazzawi, A. K., Rais, H. M., & Basri, S. (2019, April). Hybrid Artificial Bee Colony Algorithm for t-Way Interaction Test Suite Generation. In *Computer Science On-line Conference* (pp. 192-199). Springer, Cham.

Alsariera, Y. A., & Zamli, K. Z. (2015). A bat-inspired strategy for t-way interaction testing. *Advanced Science Letters*, *21*(7), 2281-2284.

Alsariera, Y. A., & Zamli, K. Z. (2015). A bat-inspired strategy for t-way interaction testing. *Advanced Science Letters*, *21*(7), 2281-2284.

Alsariera, Y. A., Ahmed, H. A. S., Alamri, H. S., Majid, M. A., & Zamli, K. Z. (2018). A bat-inspired testing strategy for generating constraints pairwise test

suite. *Advanced Science Letters*, *24*(10), 7245-7250.

Alsariera, Y. A., Majid, M. A., & Zamli, K. Z. (2015). Adopting the bat-inspired algorithm for interaction testing. In *The 8th edition of annual conference for software testing* (Vol. 14).

Alsariera, Y. A., Nasser, A., & Zamli, K. Z. (2016). Benchmarking of Bat-inspired interaction testing strategy. *International Journal of Computer Science and Information Engineering (IJCSIE)*, *7*(71-9).

Al-Sewari, A. A., & Zamli, K. Z. (2014). An orchestrated survey on t-way test case generation strategies based on optimization algorithms. In *The 8th International Conference on Robotic, Vision, Signal Processing & Power Applications* (pp. 255-263). Springer, Singapore.

Al-Sewari, A. A., & Zamli, K. Z. (2014). An orchestrated survey on t-way test case generation strategies based on optimization algorithms. In *The 8th International Conference on Robotic, Vision, Signal Processing & Power Applications* (pp. 255-263). Springer, Singapore.

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

Banharnsakun, A., Achalakul, T., & Sirinaovakul, B. (2011). The best-so-far selection in artificial bee colony algorithm. *Applied Soft Computing*, *11*(2), 2888-2901.

Bertolino, A. (2007, May). Software testing research: Achievements, challenges, dreams. In *Future of Software Engineering (FOSE'07)* (pp. 85-103). IEEE.

Cohen, M. B., Gibbons, P. B., Mugridge, W. B., Colbourn, C. J., & Collofello, J. S.

(2003, November). A variable strength interaction testing of components. In *Proceedings 27th Annual International Computer Software and Applications Conference. COMPAC 2003* (pp. 413-418). IEEE.

Colbourn, C. J., Cohen, M. B., & Turban, R. (2004, February). A deterministic density algorithm for pairwise interaction coverage. In *IASTED Conf. on Software Engineering* (pp. 345-352).

Flores, P., & Cheon, Y. (2011, June). PWiseGen: Generating test cases for pairwise testing using genetic algorithms. In *2011 IEEE International Conference on Computer Science and Automation Engineering* (Vol. 2, pp. 747-752). IEEE.

Gao, W. F., Huang, L. L., Liu, S. Y., & Dai, C. (2015). Artificial bee colony algorithm based on information learning. *IEEE transactions on cybernetics*, *45*(12), 2827-2839.

Gao, W., & Liu, S. (2011). Improved artificial bee colony algorithm for global optimization. *Information Processing Letters*, *111*(17), 871-882.

Hervieu, A., Marijan, D., Gotlieb, A., & Baudry, B. (2016). Practical minimization of pairwise-covering test configurations using constraint programming. *Information and Software Technology*, *71*, 129-146.

Homaid, A. A. B., Alsewari, A. A., Alazzawi, A. K., & Zamli, K. Z. (2018). A kidney algorithm for pairwise test suite generation. *Advanced Science Letters*, *24*(10), 7284-7289.

Kang, F., Li, J., & Ma, Z. (2011). Rosenbrock artificial bee colony algorithm for accurate global optimization of numerical functions. *Information Sciences*, *181*(16), 3508-3531.

Karaboga, D., & Akay, B. (2009). A comparative study of artificial bee colony algorithm. *Applied mathematics and computation*, *214*(1), 108-132.

Karaboga, D., & Akay, B. (2011). A modified artificial bee colony (ABC) algorithm for constrained optimization problems. *Applied soft computing*, *11*(3), 3021-3031.

Karaboga, D., & Ozturk, C. (2010). Fuzzy clustering with artificial bee colony algorithm. *Scientific research and Essays*, *5*(14), 1899-1902.

Karaboga, D., & Ozturk, C. (2011). A novel clustering approach: Artificial Bee Colony (ABC) algorithm. *Applied soft computing*, *11*(1), 652-657.

Kashan, M. H., Nahavandi, N., & Kashan, A. H. (2012). DisABC: A new artificial bee colony algorithm for binary optimization. *Applied Soft Computing*, *12*(1), 342-352.

Khalsa, S. K., & Labiche, Y. (2014, November). An orchestrated survey of available algorithms and tools for combinatorial testing. In *2014 IEEE 25th International Symposium on Software Reliability Engineering* (pp. 323-334). IEEE.

Kuhn, D. R., & Reilly, M. J. (2002, December). An investigation of the applicability of design of experiments to software testing. In *27th Annual NASA Goddard/IEEE Software Engineering Workshop, 2002. Proceedings.* (pp. 91-95). IEEE.

Kuhn, D. R., Bryce, R., Duan, F., Ghandehari, L. S., Lei, Y., & Kacker, R. N. (2015). Combinatorial testing: Theory and practice. In *Advances in Computers* (Vol. 99, pp. 1-66). Elsevier.

Kuhn, D. R., Wallace, D. R., & Gallo, A. M. (2004). Software fault interactions and implications for software testing. *IEEE transactions on software engineering*, *30*(6), 418-421.

Li, G., Niu, P., & Xiao, X. (2012). Development and investigation of efficient artificial bee colony algorithm for numerical function optimization. *Applied soft computing*, *12*(1), 320-332.

Li, J. Q., Xie, S. X., Pan, Q. K., & Wang, S. (2011). A hybrid artificial bee colony algorithm for flexible job shop scheduling problems. *International Journal of Computers Communications & Control*, *6*(2), 286-296.

Mezura-Montes, E., & Velez-Koeppel, R. E. (2010, July). Elitist artificial bee colony for constrained real-parameter optimization. In *IEEE congress on evolutionary computation* (pp. 1-8). IEEE.

Nasser, A. B., Alsewari, A. A., Tairan, N. M., & Zamli, K. Z. (2017). Pairwise test data generation based on flower pollination algorithm. *Malaysian Journal of Computer Science*, *30*(3), 242-257.

Nasser, A. B., Sariera, Y. A., Alsewari, A. R. A., & Zamli, K. Z. (2015). A cuckoo search based pairwise strategy for combinatorial testing problem. *Journal of Theoretical and Applied Information Technology*, *82*(1), 154.

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

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

Pan, Q. K., Tasgetiren, M. F., Suganthan, P. N., & Chua, T. J. (2011). A discrete artificial bee colony algorithm for the lot-streaming flow shop scheduling problem. *Information sciences*, *181*(12), 2455-2468.

Rabbi, K., Mamun, Q., & Islam, M. R. (2015, June). An efficient particle swarm intelligence based strategy to generate optimum test data in t-way testing. In *2015 IEEE 10th Conference on Industrial Electronics and Applications (ICIEA)* (pp. 123-128). IEEE.

Sabat, S. L., Udgata, S. K., & Abraham, A. (2010). Artificial bee colony algorithm for small signal model parameter extraction of MESFET. *Engineering Applications of Artificial Intelligence*, *23*(5), 689-694.

Shiba, T., Tsuchiya, T., & Kikuno, T. (2004, September). 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.* (pp. 72-77). IEEE.

Szeto, W. Y., Wu, Y., & Ho, S. C. (2011). An artificial bee colony algorithm for the capacitated vehicle routing problem. *European Journal of Operational Research*, *215*(1), 126-135.

Wu, B., Qian, C., Ni, W., & Fan, S. (2012). Hybrid harmony search and artificial bee colony algorithm for global optimization problems. *Computers & Mathematics with Applications*, *64*(8), 2621-2634.

Xu, Y., Fan, P., & Yuan, L. (2013). A simple and efficient artificial bee colony algorithm. *Mathematical Problems in Engineering*, *2013*.

Yeh, W. C., & Hsieh, T. J. (2011). Solving reliability redundancy allocation problems using an artificial bee colony algorithm. *Computers & Operations Research*, *38*(11), 1465-1473.

Yilmaz, C., Fouche, S., Cohen, M. B., Porter, A., Demiroz, G., & Koc, U. (2013).

Moving forward with combinatorial interaction testing. *Computer*, *47*(2), 37-45.

Zamli, K. Z., Din, F., Baharom, S., & Ahmed, B. S. (2017). Fuzzy adaptive teaching learning-based optimization strategy for the problem of generating mixed strength t-way test suites. *Engineering Applications of Artificial Intelligence*, *59*, 35-50.

Zhang, C., Ouyang, D., & Ning, J. (2010). An artificial bee colony approach for clustering. *Expert systems with applications*, *37*(7), 4761-4767.

Zhu, G., & Kwong, S. (2010). Gbest-guided artificial bee colony algorithm for numerical function optimization. *Applied mathematics and computation*, *217*(7), 3166-3173.