Indonesian Journal of Electrical Engineering and Computer Science
Vol. 22, No. 3, June 2021, pp. 1716~1722
ISSN: 2502-4752, DOI: 10.11591/ijeecs.v22.i3.pp1716-1722

Hybrid swarm intelligence-based software testing techniques for improving quality of component based software

Palak, Preeti Gulia, Nasib Singh Gill
Department of Computer Science and Applications, Maharshi Dayanand University, Rohtak, India

Article Info

Article history:

Received Mar 16, 2021
Revised Apr 27, 2021
Accepted May 1, 2021

Keywords:

Artificial bee colony
Automation testing
Metaheuristics
Soft computing
Test case selection

ABSTRACT

Being a time-consuming and costly activity, software testing always demands optimization and automation. Software testing is an important activity to achieve quality and customer satisfaction. This paper presents a comparative evaluation of different hybrid automated software testing techniques using the concepts of soft computing for overall quality enhancement. A comparison between three hybrid automation techniques is carried out i.e., hybrid ant colony optimization-genetic algorithms (ACO-GA), hybrid artificial bee colony (ABC)-Naïve Bayes, hybrid ABC-GA along with three parent approaches. The comparison is made by applying these hybrid techniques for the selection of minimized test suites thus reducing overall testing effort and eliminating useless or redundant test cases. The experimental results prove the efficiency of these hybrid approaches in different scenarios. The impact of automated testing techniques for quality enhancement is assessed in terms of defect density and defect detection percentage.

Corresponding Author:

Palak
Department of Computer Science and Applications
Maharshi Dayanand University
Rohtak, India
Email: palak.aug6@gmail.com

1. INTRODUCTION

In this digital era, high quality software is the need of the hour. Customers expectation demands fault free and high performing system. The dependence of industry and household on software driven devices is increasing exponentially. The customer's expectations towards software quality are also rising due to rising competition in the market. The concept of smart homes and smart cities is highly influential that can transform the face of the future. Technology has become a part of basic need of human life. Whenever there is a demand of high-quality software, the first thing that come into mind is the testing. Software testing mainly intents to reveal the defects and possible faults in the system. Testing activities consumes a substantial amount of time and money investment from overall budget of software organizations. The defects found at earlier stages of development proves less costlier than those found at later stages. The more the age of a defect, the more blunders it can create. Quality being the prime aspect of customers' satisfaction, more and more work needs to be done in this area. Based on the content analysis of various websites, tester's knowledge, documentation, maintainability, effectiveness, and reusability. are important factors for test case quality [1]. With the shift in the development paradigm from procedural approach to modular component-based approach, the overall expectations from finally delivered product are highly raised. Standardized process at each phase of development model results into high quality products. Test automation largely impacts overall quality of the software. Over years, many practitioners have employed various metaheuristic

techniques for automation of software testing [2]. Testing effort consumes a large portion of valuable resources which should be effectively managed as much as possible without compromising the effectiveness. Testing time can be as low as sixty seconds [3] but may increase exponentially with changing requirements.

With the increasing complexity of software, the need for automation testing automatically comes into mind that involves writing test scripts, running them on the programs to be tested and comparing the actual and desired outputs. The benefit of automation testing lies into the fact that the test scripts are easily modifiable and reusable. Automation can be achieved at various phases of testing life cycle. "Test case prioritization (ordering)" [4] is one such activity that has attracted many researchers in object-oriented and component-based paradigm. Metaheuristics and search-based optimization techniques are well explored for optimization of testing process including test case generation, prioritization and selection without compromising with the quality and effectiveness of the test suite. Inspired from natural behaviour of swarm intelligence of living organisms in their natural habitat, these soft computing approaches are capable of providing near optimal solution within given time constraint. Two important competitive metaheuristics technique that are very popular among tester's community are ant colony optimization (ACO) [5] and ABC (artificial bee colony) optimization [6]. Besides these algorithms, the evolutionary technique i.e., genetic algorithm (GA) enjoys very vast applications area in the field of optimization [7], [8] including test case selection. Effectiveness of test suits can be predicted in terms of number of mutants killed and total coverage achieved [9]. This article compares three different variations of modified hybrid techniques that are based on metaheuristics for assessing their effectiveness in achieving higher coverage and early detection of faults.

2. RELATED WORK

This section summarizes some important findings in the related field available in vast literature in the form of books, journals, and proceedings. Over last decade, the application area of "nature inspired optimization techniques" is expanding with the inherent benefit of providing near optimal solution in feasible amount of time. Moreover, the original techniques are now further exploited by hybridizing them with others to get the maximum benefit and optimization in consumption of resources.

Being a powerful yet simple technique, ABC is employed by researchers for optimization of testing process and also for automation. ABC is well explored in the field of software testing over past decade by augmenting it with other techniques to enhance its performance. To make best use of it and to achieve highest efficiency several hybrid methods have been proposed over years which resulted into vast literature available online and offline. ABC and its modified versions are exploited in various fields of engineering [10]-[14] which depicts the overall effectiveness and versatility of the technique.

Some researchers compared and assessed the applications of various metaheuristics techniques for optimization of testing process and argued that swarm-based techniques perform better for the given scenario [15]. Various evolutionary techniques are used for path testing for achieving higher coverage [16]. Omur Sahin et. Al. made a comparison of various metaheuristics algorithms for "test data generation" and their experimental results proves that ABC performs superior in case of large search space [17].

Cuckoo search has been applied to a wide range of applications including software testing. The same is applied along with bee colony on model based testing for automated test case generation and selection [18]. The authors utilized benefits of both optimizers for software testing using state chart and sequence diagram of the system under test. They considered automated teller machine (ATM) authentication and withdrawal functionality for evaluating the performance of their proposed hybrid technique. The "Levy flight" from cuckoo search method is exploited to search the best nest and abandon the others. Various modifications in original ABC are carried out over years. One such example is "Scout less ABC with modified onlooker bees". It was argued by the authors that scout bees are counterproductive. To reach more diversity among population, onlooker bees are modified and the performance is evaluated using well known classification problems by training fuzzy neural network [19]. This methodology provides a light weight optimizer for test case optimization. State table-based testing is also gaining importance day by day. A "comprehensive improved ACO" is proposed on state table that is generated from state graph. ACO is applied to achieve high coverage and efficiency [20]. The native exploration of ABC algorithm lacks an information-based procedure [21]. So, to improve the global information-based solution, memory element is added to employed bees that remembers the global best solution so far. This provides a hybrid ABC approach that is based on PSO's gbest and pbest parameters [22]. A similar approach for optimization that is based on "best neighbor guided ABC" has also been proposed. They argued that it is risky to merely abandon the exhausted food source at scout bee phase, because the discarded one may have more beneficial information. The authors compared the results of their proposed technique with several other variants of ABC and achieved higher efficiency. Some reaserchers focused more on data flow testing which is a white box testing technique and utilized the concept of memory based ABC for path saving and achieved overall test suite

saving [23]. Thus, it can be clinched that hybrid ABC with memory element for saving the global best solution has gained more popularity and shown effective results. Various swarm intelligence techniques for regression test case selection over benchmark problems are been compared. Through experimental results they concluded that hybrid PSO outperforms ACO with 0.7 % test case selection [24]. Researchers employed ABC for path coverage by finding optimal fitness value among a range of values [25]. A hybrid ACO-GA (HACGA) approach for test case selection from fault matrix is presented which is capable of overall saving in execution time and effort by selecting a minimized set of test cases [26]. The authors also proposed a technique for reducing the size of test suite by utilizing the benefits of modified ABC and Naïve Bayes classifier over component based projects [27]. Another approach that is based on scout-less ABC and GA is implemented on some well-known problems for test case selection provides significant saving in execution time and removal of redundant test cases [28].

3. RESEARCH METHOD

This research mainly focuses on the quality improvement of component-based software by applying hybrid soft computing-based automation testing techniques. A comparison is made between three parent approaches i.e., GA, ACO, ABC and three hybrid versions i.e., ACO_GA (HACGA) [24], hybrid ABC_Naive Bayes [25], hybrid ABC_GA [26] and their performance is evaluated in terms of execution time, percentage of test cases selected and average branch coverage.

There are various ways of validating the comparison results. In this study, we considered a working example of "Simple Word Processor" (a windows form application) developed in C#.Net using component-based paradigm to validate the comparison results. The project under consideration consists of six components and sixteen sub-components. The main six components of this project are: File, Edit, Font, Paragraph, Insert Image and Color. The sub-components in each category are listed in Table 1. Each sub-component of the project is injected with some known defects to test the efficiency of selected test cases after applying the hybrid automation techniques.

Table 1. Details of project under test: components, sub-components and defects injected

Project Under Test	Components	Sub-components	No. of Defects Injected
Simple Word Processor	File	Open	3
		New	4
		Save	3
		Print	5
	Edit	Cut	4
		Copy	5
		Paste	4
		Select All	3
		Find and replace	4
	Font	Font Face	4
		Font Style	3
		Font Size	3
	Paragraph	Indent	3
		Align	4
	Insert Image	Browse and upload image	5
	Color	Select Color	3
			Total Defects Injected=60

4. RESULTS AND DISCUSSION

The hybrid nature-inspired techniques are compared on the basis of three major parameters taken into consideration i.e., execution time, percentage of test case selected and average branch coverage in context of software testing. A comparison between three hybrid automation techniques is carried out i.e., hybrid ACO-GA, hybrid ABC-Naïve Bayes, hybrid ABC-GA. The experimental results proves that all the three techniques perform better in given scenario in comparison to original non hybrid approaches. Hybrid ACO-GA works better to its full capacity for smaller fault matrix and lower complexity projects. Hybrid ABC-Naïve bayes uses the concepts of classification for further enhancement of solution space. This approach is better than hybrid ACO-GA and capable of handling larger complex problem domain. Hybrid ABC_GA works comparable to hybrid ABC_Naïve Bayes for higher fitness function value range.

4.1. Execution time

Figure 1 shows a comparative evaluation of the experimental results over increasing size of fault matrix. It is found that the execution time of GA increases abruptly with increase in fault matrix size. So, it is

not suitable for large data set. The performance of hybrid ACO-GA also degrades with increase in fault matrix size. While hybrid ABC_Naïve Bayes is the winner in terms of execution time and performance trade off. The lower the execution time, the better and faster the technique is. Thus, hybrid techniques offer huge amount of time saving as compared to the original soft computing techniques.

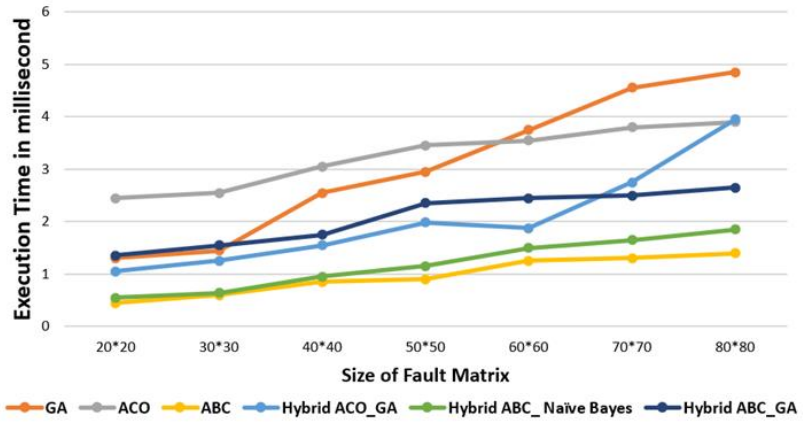

Figure 1. Comparison of execution time

4.2. Percentage of test case selected

The above mentioned six approaches are exploited for selection of test cases among seven different sized fault matrices i.e., 20 by 20, 30 by 30, 40 by 40, 50 by 50, 60 by 60, 70 by 70 and 80 by 80. With increase in the fault matrix size, the performance of the techniques under consideration also gets affected. Figure 2 shows the experimental results in graphical form. Here, GA is the least scorer, while hybrid approaches win the race. The hybrid ABC_Naïve Bayes and hybrid_GA are capable of selecting an efficient test suite as low as 32% of the total test cases which in turn improves overall quality and performance.

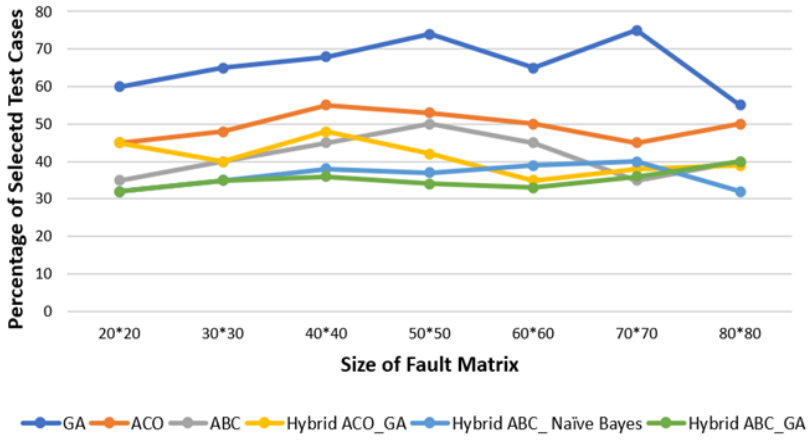

Figure 2. Percentage of test case selected

4.3. Comparison of average branch coverage

Branch coverage is an important aspect of testing the effectiveness of selected test cases. Higher the branches covered by the test suite, the higher is its efficiency to detect faults that occur at branching point. Branching is a result of either a control statement that looks for a condition to be satisfied or due to looping. In this research, the above mentioned six automation approaches are evaluated for 10 to 80 number of cycles with a step size of ten. The results are represented in Figure 3 for comparison. It was deduced that original GA underperforms due to various inherent disadvantages while the hybrid approaches provide higher average branch coverage. The pattern also revealed a positive correlation between number of cycles and average branch coverage. The less complex projects with Max. Cyclomatic Complexity < 5 are best tested using

hybrid ACO_GA and hybrid_Naive Bayes while the projects with higher complexity (i.e., complexity > 5) are suitable to be tested using hybrid ABC_GA.

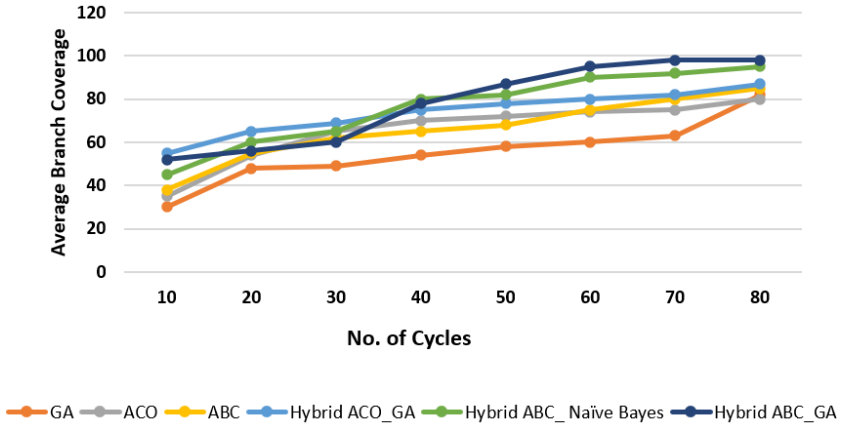

Figure 3. Comparison of average branch coverage

4.3.1. Impact of hybrid automation testing on software quality

For quantitative evaluation of software's quality, ISO 9126-1 [27] standardized set of six quality characteristics: "Functionality, Reliability, Usability, Efficiency, Maintainability and Portability". These six characteristics are further divided into various sub- characteristics. Later in 2011, this international standard has been replaced by ISO/IEC 25010 which defined "eight characteristics" adding "security" and "compatibility" to the previous one. This research mainly provides an enhancement in three main quality characteristics: Functionality, Reliability and Maintainability which conspicuously affect the quality of the software at large. By functionality, we mean the correct and appropriate functional behaviour of all the components that consist the system. Whereas, reliability is related to fault free system availability. Maintainability in broad sense can be defined in terms of system's ability towards seamless changes, testability and stability. Higher the system modularity, the higher the maintainability is. Although reliability and maintainability consist of many sub-factors within themselves and are very difficult to measure, we use simple metric for their assessment. The three hybrid techniques mentioned in this article are applied as after the another and average value is taken into consideration which is listed in Table 2. An important metric for measuring the impact of automated hybrid testing techniques on quality enhancement is "defect density (DD)" and "defect detection percentage (DDP)". The experimentation results prove the positive impact of hybrid testing techniques on above mentioned quality factors. Test automation largely saves time and efforts of the development team. The hybrid automation approaches are additional rewards to the testers' community.

Table 2. Impact of hybrid testing techniques on quality parameters

Test Case Selection Technique	Total Number of Defects Injected	#defects detected	Defect Detection Percentage	Actual Defect Density per component	Observed Defect Density per component
GA	60	40	66.7	10	6.7
ACO	60	42	70	10	7
ABC	60	45	75	10	7.5
Hybrid ACO_ GA	60	58	96.7	10	9.7
Hybrid ABC_Naïve Bayes	60	58	96.7	10	9.7
Hybrid ABC_GA	60	59	98.3	10	9.8

4.3.2. Merits and demerits of nature inspired optimization techniques

In this section, a summarized view of merits and demerits of comparative techniques is presented. During this research, it was found that GA is simple to implement, but in terms of execution time, there are no standardized way to predict the same. ACO and ABC depicts parallel behaviour but this behaviour is unpredictable in same of large data set. Moreover, convergance time is also unpredictable. Most of the previous original metaheuristic techniques lack control over parameter tuning and the practitioner need to use "hit and trial" method for optimizing their results as per the application area. The other important drawback that can be

challenged is the problem of "getting stuck to the local optima" and losing the power to utilize full potential of the population diversity. While using the hybrid approaches, we have a tradeoff between execution time and efficiency. With little more efforts spent on control parameters, we can get better results in terms of efficient test case selection. Table 3 presents the detailed findings of pros and cons of each approach.

Table 3. Merits and demerits of nature inspired optimization techniques

Hybrid Approach	Merits	Demerits
GA	GA is simple to implement. The operators used are very easy to modify. GA can be used to solve large space problems and non- linear functions.	GA may get stuck to local optima. GA has no standardized way for defining fitness value. GA alone is not capable of utilizing full population diversity. GA is not effective in terms of execution time for large problem space.
ACO	Parallel and simultaneous approach Avoids premature convergence	Convergence time is undefined Random behavior of ants is unpredictable in some cases Lower success ratios
ABC	Robust Flexible and simple Very few controls parameters Parallel approach	Premature convergence in large domain problems Classification accuracy may degrade with increase in sample size.
Hybrid ACO_GA	Parallel approach Better diversification	Higher Complexity. Slower than its parent approaches.
Hybrid ABC_Naïve Bayes	Faster Very few controls parameters Parallel and robust approaches Classification accuracy is not affected by sample size Parallel Approach	Performance degrades at higher complexity
Hybrid ABC_GA	Flexible with few controls parameters Parallel Approach	Execution time may increase with larger data sets.

5. CONCLUSION

Software quality is the prime aspect of any industry for achieving long term business goals. Product quality is largely affected by development process quality. Efficient software testing is capable of controlling few important software quality characteristics namely: Functionality, Reliability and Maintainability. In this article, six nature inspired optimization techniques are evaluated in terms of test case selection automation, three out of which are hybrid approaches namely hybrid ACO_GA, Hybrid ABC_Naive Bayes and Hybrid ABC_GA. These techniques are applied on working project that is developed using component-based paradigm. A total of sixty defects were injected at various components of this project to test the effectiveness of selected test suite. It can be argued from the experimental results, that the hybrid techniques performs better than their parent algorithms with minimum trade-off. Their impact on software quality is also considered and evaluated. Hybrid testing techniques provide promising results and act as a helping tool to the testers' community by saving time and overall cost thereby increasing customer's satisfaction and overall quality.

ACKNOWLEDGEMENTS

This research work was supported by Department of Computer Science and Applications, Maharshi Dayanand University, Rohtak, India. The authors are grateful to all the staff and students of this institution for their assistance.

REFERENCES

[1] S. O. Barraood, H. M. Haslina, F. Baharom, and M. Intelligences, "Test Case Quality Factors : Content Analysis of Software Testing Websites," *Webology Spec. Issue Artif. Intell. Cloud Comput.*, vol. 18, pp. 75–87, 2021, doi: 10.14704/WEB/V18SI01/WEB18007.

[2] M. Mann, P. Tomar, O. P. Sangwan, and S. Singh, "Automated Software Testing using Metahurestic Search : A Critical Review," *Electron. J. Biol.*, vol. 12, no. 2, pp. 145–155, 2016.

[3] I. Mahmud, F. Sadia, M. Rahman, S. Ahmed, and D. Islam, "Web usability test in 60 seconds: A theoretical foundation and empirical test," *Indonesian Journal of Electrical Engineering and Computer Science (IJEECS)*, vol. 17, no. 1, pp. 398–403, 2019, doi: 10.11591/ijeecs.v17.i1.pp398-403.

[4] U. Farooq, H. Aman, A. Mustapha, and Z. Saringat, "A review of object-oriented approach for test case prioritization," *Indonesian Journal of Electrical Engineering and Computer Science (IJEECS)*, vol. 16, no. 1, pp. 429–434, 2019, doi: 10.11591/ijeecs.v16.i1.pp429-434.

[5] M. S. A. Forhad, M. S. Hossain, M. O. Rahman, M. M. Rahaman, M. M. Haque, and M. K. H. Patwary, "An improved fitness function for automated cryptanalysis using genetic algorithm," *Indonesian Journal of Electrical Engineering and Computer Science (IJEECS)*, vol. 13, no. 2, pp. 643–648, 2019, doi: 10.11591/ijeecs.v13.i2.pp643-648.

[6] A. Yani, Junaidi, M. Irwanto, and A. H. Haziah, "Optimum reactive power to improve power factor in industry using genetic algortihm," *Indonesian Journal of Electrical Engineering and Computer Science (IJEECS)*, vol. 14, no. 2, pp. 751–757, 2019, doi: 10.11591/ijeecs.v14.i2.pp751-757.
[7] M. Akour and M. Alenezi, "Test suites effectiveness evolution in open source systems: Empirical study," *Indonesian Journal of Electrical Engineering and Computer Science (IJEECS)*, vol. 19, no. 2, pp. 992–999, 2020, doi: 10.11591/ijeecs.v19.i2.pp992-999.
[8] G. A. Sultan and M. K. Jarjes, "Optimal PID controller design using artificial bee colony algorithm for robot arm," *Indonesian Journal of Electrical Engineering and Computer Science (IJEECS)*, vol. 21, no. 1, pp. 84-91, 2021, doi: 10.11591/ijeecs.v21.i1.pp84-91.
[9] C. Hernández, J. Rodríguez, and D. Giral, "Spectrum allocation model for cognitive wireless networks based on the artificial bee colony algorithm," *Indonesian Journal of Electrical Engineering and Computer Science (IJEECS)*, vol. 19, no. 1, pp. 257–266, 2020, doi: 10.11591/ijeecs.v19.i1.pp257-266.
[10] A. M. Abdulazeez, D. M. Hajy, D. Q. Zeebaree, and D. A. Zebari, "Robust watermarking scheme based LWT and SVD using artificial bee colony optimization," *Indonesian Journal of Electrical Engineering and Computer Science (IJEECS)*, vol. 21, no. 2, pp. 1218–1229, 2021, doi: 10.11591/ijeecs.v21.i2.pp1218-1229.
[11] P. C. Saibabu, H. Sai, S. Yadav, and C. R. Srinivasan, "Synthesis of model predictive controller for an identified model of MIMO process," *Indonesian Journal of Electrical Engineering and Computer Science (IJEECS)*, vol. 17, no. 2, pp. 941–949, 2020, doi: 10.11591/ijeecs.v17.i2.pp941-949.
[12] K. L. Anitha and T. R. G. Nair, "Online cloud performance testing in social networks at peak demand scenarios," *Indonesian Journal of Electrical Engineering and Computer Science (IJEECS)*, vol. 17, no. 1, pp. 372–378, 2020, doi: 10.11591/ijeecs.v17.i1.pp372-378.
[13] R. R. Sahoo and M. Ray, "Metaheuristic techniques for test case generation: a review," Journal of Information Technology Research (JITR), vol. 11, no. 1, pp. 158-171, 2018, doi: 10.4018/JITR.2018010110.
[14] D. B. Mishra, A. A. Acharya, and R. Mishra, "Evolutionary algorithms for path coverage test data generation and optimization: A review," *Indonesian Journal of Electrical Engineering and Computer Science (IJEECS)*, vol. 15, no. 1, pp. 504-510, 2019, doi: 10.11591/ijeecs.v15.i1.pp504-510.
[15] O. Sahin and B. Akay, "Comparisons of metaheuristic algorithms and fitness functions on software test data generation," *Appl. Soft Comput.*, vol. 49, pp. 1202–1214, 2016, doi: 10.1016/j.asoc.2016.09.045.
[16] P. Lakshminarayana and T. V. SureshKumar, "Automatic Generation and Optimization of Test case using Hybrid Cuckoo Search and Bee Colony Algorithm," *J. Intell. Syst.*, vol. 30, no. 1, pp. 59-72, 2021, doi: 10.1515/jisys-2019-0051.
[17] K. Hussain, M. N. Mohd Salleh, S. Cheng, Y. Shi, and R. Naseem, "Artificial bee colony algorithm: A component-wise analysis using diversity measurement," *J. King Saud Univ.-Comput. Inf. Sci.*, vol. 32, no. 7, pp. 794-808, 2020, doi: 10.1016/j.jksuci.2018.09.017.
[18] S. Sankar S and V. Chandra S S, "An Ant Colony Optimization Algorithm Based Automated Generation of Software Test Cases," *International Conference on Swarm Intelligence. (ICSI 2020)*, 2020, pp. 231-239, doi: 10.1007/978-3-030-53956-6.
[19] A. K. Alazzawi, H. M. Rais, and S. Basri, "HABC: Hybrid artificial bee colony for generating variable T-way test sets," *J. Eng. Sci. Technol.*, vol. 15, no. 2, pp. 746-767, 2020.
[20] A. K. Alazzawi, H. M. Rais, S. Basri and Y. A. Alsariera, "PhABC: A Hybrid Artificial Bee Colony Strategy for Pairwise test suite Generation with Constraints Support," *2019 IEEE Student Conference on Research and Development (SCOReD)*, 2019, pp. 106-111, doi: 10.1109/SCORED.2019.8896324.
[21] S. Sheoran, N. Mittal, and A. Gelbukh, "Artificial bee colony algorithm in data flow testing for optimal test suite generation," *International Journal of System Assurance Engineering and Management*, vol. 11, pp. 340–349, 2020, doi: 10.1007/s13198-019-00862-1.
[22] A. Agrawal and A. Kaur, "A Comprehensive Comparison of Ant Colony and Hybrid Particle Swarm Optimization Algorithms Through Test Case Selection," *Satapathy S., Bhateja V., Raju K., Janakiramaiah B. (eds) Data Engineering and Intelligent Computing. Advances in Intelligent Systems and Computing*, Singapore: Springer, 2018, pp. 397–405, doi: 10.1007/978-981-10-3223-3.
[23] F. Hamad, "Using Artificial Bee Colony Algorithm for Test Data Generation and Path Testing Coverage," *Mod. Appl. Sci.*, vol. 12, no. 7, pp. 99-112, 2018, doi: 10.5539/mas.v12n7p99.
[24] Palak and P. Gulia, "Hybrid swarm and GA based approach for software test case selection," *International Journal of Electrical and Computer Engineering (IJECE)*, vol. 9, no. 6, pp. 4898-4903, 2019, doi: 10.11591/ijece.v9i6.pp49898-4903.
[25] Palak, P. Gulia, and N. S. Gill, "An Enhanced Artificial Bee Colony : Naïve Bayes Technique for Optimizing Software Testing," *Int. J. Adv. Comput. Sci. Appl.*, vol. 12, no. 2, pp. 220–225, 2021, doi: 10.14569/IJACSA.2021.0120228.
[26] Palak, P. Gulia, and N. S. Gill, "Optimized Test Case Selection using Scout-less Hybrid Artificial Bee Colony Approach and Crossover Operator," *Int. J. Eng. Trends Technol.*, vol. 69, no. 3, pp. 39-45, 2021, doi: 10.14445/22315381/IJETT-V69I3P208.
[27] *Softw. Eng. Qual. 1 Qual. Model. Geneva, Switz. Int. Organ. Stand.*, ISO IEC 9126-1, (2001).