ISSN 2278-3091
Volume 9, No.5, September - October 2020
International Journal of Advanced Trends in Computer Science and Engineering
Available Online at http://www.warse.org/IJATCSE/static/pdf/file/ijatcse207952020.pdf
https://doi.org/10.30534/ijatcse/2020/207952020

Artificial Bee Colony Based Prioritization Algorithm for Test Case Prioritization Problem

Richa Vats[1] Arvind Kumar[1]
[1]SRM University Delhi-NCR, Sonepat (Haryana), 131029, India
ritzi1606@gmail.com, k.arvind33@gmail.com

In software engineering filed, regression testing can be described as one of important task to test the testcases. In regression testing, test cases from test suites are executed repeatedly such that modifications incorporated in software cannot lead to unpredictable outcomes. As, the functionality increases in software's, the size of test suites also increased. In turn, regression testing becomes one of time consuming and costly task. So, to reduce cost and time, instead of complete test suite can run, a subset of test cases is selected. The selection of this subset is known as test case prioritization (TCP). TCP becomes the NP-hard problem when size of test suites and test cases increases. Hence, there is a need of prioritization algorithm that can select optimal subset of test cases to overcome the aforementioned problems. Recent studies showed that artificial bee colony (ABC is one of effective technique that can be solved the diverse optimization problem. Hence, ABC based prioritization algorithm is proposed to choose optimal subset of test cases. The results of ABC based prioritization algorithm are compared with sequential and random ordering. Results confirmed that it can address the TCP problem in competent manner.

Key words: Regression Testing, Test Cases, Test Suites, Artificial Bee Colony, Prioritization Algorithm

1. INTRODUCTION

In present time, software's become an essential part of everyday life and these software's are tested prior to release such that the outcomes of software can be predict. To test the software's, some test cases are designed in the form of test suites and software's are tested against each test cases. This process is called regression testing. The aim of this testing is to identify the bugs in software's. The bugs are presented in every module of software's. So, the test cases can be so efficient such that these bugs can be identified during testing process. After releasing of software's, new patches can be added into software's time to time to enhance its capabilities. When, a patch is added into software, the software can be tested again. The entire software can be re-evaluated using test cases to predict its outcomes. As the size of software evolves, the test cases can be increased and the software should be executed against each test case. Due to this, cost, time and coverage parameters of testing process can be enhanced and overall cost of software is increased[1]. The solution of this situation is to choose a subset of test cases rather than whole test suites and software will be tested against this subset of test cases.But, task to determine the sequence of test casesfor testing the software can be considered as tough task and bugs can be identified successfully [2]. This process is also known as test case prioritization and it becomes NP-hard problem when size of test suites can increase.In TCP, test cases are arranged according to some priority. The priority is computed using some objective function [3, 4]. Furthermore, coverage and detection of faultscan be used to describe objective function and worked as a priority. This value is associated with each test cases and these are executed according to the priority value. High priority test cases executed first and so on. The advantage of prioritization is to collect earlier feedback and this feedback can make the debugging process easy for developers.

Further, it is noticed that prioritization of testsare characterized on the statement of coverage and faults. Moreover, a fitness criterion is also defined for prioritization. On the basis of fitness criterion values, test cases are selected from test suites. In turn, the prioritization can be viewed as multivariable optimization problem [5].

In literature, it is found that large numbers of meta-heuristic techniques adopted foraddressing the prioritization problem. These meta-heuristic techniques obtain optimal and near optimal solution for TCP problem. The several meta-heuristic techniques that can be successfully applied for prioritization of test case are Tabu Search [6], Hill climbing [7], and Genetic Algorithm (GA) [7]. But there is always a scope to enhance the results of optimization techniques and none of technique can solve a problem with hundred percent accuracy rates.

1.1 Contribution of this work

This works presents an efficient and effective prioritization algorithm for selecting the optimum set of test-cases. It is noticed that ABC algorithm provides optimum results for many optimization problems. So, an ABC based prioritization algorithm is proposed for test case prioritization. Further, few improvements are also inculcated in ABC algorithm for improving its efficacy. The experimental results of ABC prioritization are compared with GA, K-Means, PSO based prioritization algorithms.

The structure of paper is given as section 2 demonstrates the related works in the direction of prioritization algorithms. The basic ABC algorithm is described in section 3. Section 4 illustrates improvements in ABC algorithms as well as ABC based prioritization algorithm.

The experimental results of this paper are reported in section 5. The entire paper is concluded in section 6.

2. RELATED WORKS

The related works on the prioritization algorithms are highlighted as below.

Bajwa and Kaur developed an adaptive approach for test cases prioritization based on the genetic algorithm [8]. It can speed up the scheduling of test cases. To improve the coverage of test cases prioritization, an immune based genetic algorithm is reported in [9]. In the proposed approach, an immune operator is incorporated in genetic algorithm to overcome the low convergence problem. It is noted that IGA give better results than genetic algorithm. A hybrid approach based on genetic algorithm and simulated annealing is reported for TCP [10]. This approach can reduce cost as well as enhance fault rate. Tulasiraman and Kalimuthu developed a cognizant cost and history-based TCP approach [11]. The proposed approach is used the historical information of the test cases for identification of fault rate and cost. Moreover, artificial immune system algorithm is also applied to find the effective test cases. A multiobjective search-based regression TCP approach is presented in [12]. The proposed approach is the combination of the epistasis theory and ant colony optimization algorithm (ACO). The epistasis theory is used to update the pheromone strategy of ACO algorithm. To enhance the effectiveness of TCP, chen et al. [13], presented an adaptive random sequence approach. The proposed approach consists of two clustering algorithm such as K-means and K-medoid. The simulation results stated that the proposed approach enhances earlier detection of fault rate. To detect the faults earlier, a fuzzy TPOSIS technique is reported for prioritizing the test cases [14]. In this approach, fuzzy principles are used for decision making. A risk-based prioritization approach is reported for test cases [15]. In this work, fuzzy expert system is developed to accurate detection of risks or faults. Noguchi et al. [16] developed a frame work for TCP using ant colony optimization algorithm. Jiang and Chan presented local beam search-based technique for effective TCP [17]. The proposed approach is validated using four benchmarks test cases datasets and gives better results than greedy and genetic algorithms. Prioritizing the test cases based on total coverage, Konsaard and Ramingwong applied a modified genetic algorithm for TCP [18]. A greedy based prioritization approach is reported for optimizing the TCP problem [19]. The proposed approach consists of exploration strategy and multi level coverage model to capture the bugs. A multi-objective genetic algorithm is reported for TCP for reducing the cost of regression testing [20]. In this work, a mechanism based on orthogonal design and evolution is incorporated in multi objective GA. It is seen that DIV-GA is more capable than other algorithms. To optimize the test cases in time constrained environment, panwaret al. [21] presented a hybrid approach by combining CS and modified ACO algorithm for obtaining optimized test cases. A Bayesian based clustering approach is presented to prioritize test cases [22]. In this work, two java projects are considered to identify the mutated faults. The performance of the work is compared with greedy approach and BNA techniques. It is stated that Bayesian based clustering gives promising results. To detect faults with minimum time and earlier, Tulasiraman et al. [23] presented pareato and clonal selection algorithm based multi-objective approach for TCP. It is noticed that proposed multi objective approach scheduled the test cases optimally and earlier. Suri and Singhal presented ACO based technique for regression testing and prioritization [24]. Further, it is seen that a time bounded constraint is incorporated in proposed approach to determine optimal test cases. Results confirm that ACO based technique is one of effective technique for TCP. To maximize the fault coverage, Mann et al. [25] have applied PSO based prioritization algorithm to solve the TCP problem efficiently. The effectiveness of the PSO based prioritization algorithm is measured using small as well as large test suites. It is seen that the proposed prioritization algorithm can handle both of test suites effectively. To handle the prioritization task in High Configurable Systems, Parejo et al. [26] have developed a Drupal based framework. This framework consists of multi objective prioritization algorithm rather than single objective prioritization algorithm. It also noted that proposed framework can handle change in feature property of dataset in effective manner. Results indicated that the proposed multi objective prioritization algorithm gives effective results than single objective prioritization algorithm. Schwartz and DO [27] presented two cost effective prioritization techniques. These techniques are based on analytic hierarchy process and weighted sum model. The experimental results of these techniques are compared with existing cost-effective techniques. It is noticed that proposed techniques can improve the cost effectiveness of regression testing. Marchetto et al. [28]developed multi objective technique for addressing cost and coverage factors. Further, this method also prioritizes the test cases. The twenty-one java projects are adopted for evaluating the performance of proposed multi objective technique. It is observed that the proposed technique is one of competitive technique for prioritization of test cases.

3. ARTIFICIAL BEE COLONY ALGORITHM

ABC is a meta-heuristc technique inspied through bee behaviour [29]. Initially, this technique can be used for solving the function optimization problems. The working this technque can be described through EB Phase, OB Phase and SB Phase. In ABC alorithm, the optimum solution can be denoted using food source. The aim of bees is to locate the position of food source. Thus, each bee have unique ability to locate the food source position. The algorithm starts with Employed bee phase. In this phase, bee search the location of food source, collects the information regarding food and send it to next pahse. The nest phase of algorithm is Onlooker bee phase. This phase measure the quality of information collected in previous phase. If information quality is not good, then, bee search the new location of food in nearby area. The Scout bee phase is invoked, when onlooker bee is not able to improve the quality of food in using a limit operator and the location of food abandoned. The work of this phase is to determine the new location of abandoned food. The working of ABC is mentioed in Algorithm 1.

Algorithm 1: Algorithmic Steps of ABC

1. Initialization Phase
   - Initialize the different user defined parameter of ABC algorithm like population, food source, maximum iteration, limit, colony size, lower bound and upper bound.
   - iteration = 0;
2. EB Phase

   for bee_i = 1: FS
   - Update the location of food using equation 1 for employed bee phase

$$X_{i,new} = X_i + \emptyset(X_j - X_k). \quad (1)$$

   - Evaluate the fitness of newly generated food source using equation 2.

$$fit_i = \frac{1}{1 + f_i}. \quad (2)$$

   - Perform greedy selection between location foods.

3. OB Phase
   - Evaluate the probability of each food source

   for each bee_i = 1: food source

   if (rand () < P_i)
   - Determine new food source location using equation 3.

$$X_{i,new} = X_i + \emptyset(X_j - X_k). \quad (3)$$

   - Evaluate the fitness of food through equation 2.
   - Apply greedy selection between location of foods.

   else
   - bee_i=bee_i+1;

4. SB Phase

   IF (Is quality of food improved using limit operator)
   - Determine location of food through scout bee in random order.

   end if
   - Memorize best solution
   - Iteration = iteration + 1
   - Obtain final results

4. PROPOSED IMPROVED ABC

In literature, ABC is one of popular algorithm for solving wide variety of optimization problems. Further, it is noticed that exploration and exploitation processes are key concept of meta-heuristic algorithms [30]. The balance between these two processes can be maintainedfor obtaining optimum results. It is observed that exploration process of ABC algorithm is good, but lack with exploitation process [31]. Due to this, algorithm suffers with slow convergence. Moreover, the same search equation is used in EB phase andOB phase. In turn, ABC algorithm suffers with population diversity in last iterations. Hence, to make the ABC algorithm more robust and efficient, few modifications are proposed to avoid abovementioned problems.

1. A concept of pbest measure is added into food search equation of employed bee phase.
2. A concept of gbest measure is included in the food search equation of onlooker bee phase.
3. A cost-effective strategy is developed to reduce cost.

4.1 pbest and gbest Measures

These measures are taken from the PSO algorithm. The pbest measure computes the personal best position of individual bee. While, the gbest measure can determines global best position of bees. In employed bee phase, a bee searches the food location in random order using equation 1. So, in equation 1, there is no guidance about the best position of individual bee. Here, the concept of pbest is added to guide the search mechanism of individual bee. The updated food search equation for employed bee phase is

$$X_{i,new} = X_i + \emptyset(X_i - X_k) + \emptyset(pbest_i - X_k). \quad (4)$$

The gbest measure computes the global best position among all individuals. Moreover, this measure can compute the direction of optimal solution. In OB phase, food quality is evaluated and food denotes the possible solution of problem. In original ABC, same equation is adopted to search the food in both phases. In this work, the concept of gbest measure is added into food search equation of onlooker bees to direct the optimal solution.

$$X_{i,new} = X_i + \emptyset(X_i - X_k) + \emptyset(gbest - X_k). \quad (5)$$

4.2 Cost Effective Strategy

A cost-effective strategy based on weights is developed in this work. A test suits contains numbers of test cases and these test cases are derived using some criteria. Suppose, test cases are represented as $T = \{t1, t2, t3, \ldots\ldots, tn\}$ and the criteria are denoted as $C = \{ c1, c2, c3, \ldots\ldots, cn \}$. After, test cases are executed, some values are assigned to test cases regarding its effectiveness. On other hand, some values are also assigned to criteria's that are used to design the test cases. A decision matrix is performed using test cases and design criteria.It can prioritize test-cases using decision matrix.

Algorithm 2: ABC based Prioritization Algorithm

1 Randomly selects Ktest cases from test suite with M sequences.

2 Initialized other algorithmic parameters like number of food sources, limit operator=5, no. of faults, maximum iteration=100, iteration=0.

3 While(iteration <= maximum iteration)

4 EB strats

for employed_bee$_i$ = 1: FS

- Send employed bee to locate the position of food through equation 1.
- Evaluate fitness of food source through equation 6.

$$fit_i = \frac{1}{1 + f_i}. \quad (6)$$

- Determine the pbest position of each employed bee and put it pbest_pool.
- Apply Greedy selection to determine best location of food.
- Determine the gbest position.

5 OB Phase

- Evaluate probability of each food using equation 7.

for each onlooker_bee$_i$ = 1: food source

if (rand () < P_i)

- Determine new food location through equation 3.
- Evaluate fitness of food through equation 2.
- Apply greedy selection for determining best food location.

else

- bee$_i$=bee$_i$+1;

6 SB Phase

IF (Is quality of food improved through limit operator)

- Determine the new location of food through scout bee in random order.

end if

7 Memorize best solution and put in candidate pool C.

8 Iteration = iteration + 1

10 Termination condition is not reached, repeat steps 4-

11 Obtain the prioritized test cases as output.

4.3 ABC based Prioritization Algorithm

The basic steps of ABC based prioritization algorithm are described in Algorithm 2.

4.4 Complexity of Proposed Algorithm

The algorithm starts with the randomly defined population in terms of test cases. Further, the nest step is to initialize the user defined parameters of prioritization algorithm such as food source, colony size, limit operator, pbest, no. of faults and maximum iteration. In this work, APFD is considered as fitness function that can be used to evaluate the quality of food source positions. Initially, employed bees explore the food location in search area i.e. subset of test cases using equation 4. The next step is to determine the fitness of food. It is computed through equation 1. The onlooker bee phase collects the information from the previous phase. Moreover, a probability function is measured for each food and also computes the food quality i.e. set of test cases. If food quality not updated, then, an onlooker bee is sent to discover new location of foods and can be determined through equation 6. Again, fitness of recently discovered food is computed. Otherwise, onlooker bee is incremented by one. Further, make the greedy selection between the previous food and current food; and put best into a candidate pool. Is quality of food improved through limit operator, then it is abandoned. A new location of food is discovered in random manner. The algorithm stops its working, after reaching maximum iterations and the optimal set of test cases is obtained. Otherwise, above

mentioned process will be continued. The complexity of the proposed algorithm depends on test cases (N), selected test cases(K)and sequence of test cases sequence (M). After analyzing the pseudo code of proposed algorithm, the complexity can be $O(N \times K \times M)$.

5. SIMULATION ENVIRONMENT

ABC prioritization algorithm is validated using an ATM system cases study is used. The ATM application consists of five modules such as Log_In module, Pin_Change module, Balance_Enquiry module, Cash_Withdrawal module, Cash_Deposit module. The program is written in C++ language and having 140 line of code. A test suite is designed to test the five modules of ATM application. The faults and execution time are taken as performance parameter or cost of test cases. The test suite consists of ten test cases in sequential order and can be represented as $T = \{t_1, t_2, t_3, t_4, t_5, t_6, t_7, t_8, t_9, t_{10}\}$.The faults are represented using $F = \{u_1, u_2, u_3, u_4, u_5, u_6, u_7, u_8\}$. The experimental results of ABC prioritization algorithm are given Table 1. Moreover, in this work, random and original ordering to test cases is also considered for determining optimal sequence of test-cases.

Table 1:illustrates faults detection and execution time of ABC prioritization algorithm

Test Case (T)	Faults (F)								No. of Faults Detected	Execution Time
	u_1	u_2	u_3	u_4	u_5	u_6	u_7	u_8		
t_1	-	-	-	∅	-	-	∅	-	2	9.30
t_2	-	-	-	∅	-	∅	-	∅	3	10.45
t_3	-	-	∅	-	-	-	∅	-	2	8..25
t_4	∅	-		∅	-	∅	-	∅	4	9.48
t_5	-		∅	-	∅		-	∅	3	11.05
t_6	-	-	-	-	∅	∅	-	-	2	7.28
t_7		∅		∅	-	-	∅	-	3	8.46
t_8	∅	∅	∅	-	-	∅	-	∅	5	10.56
t_9	∅	-	-	-	∅	-	-	∅	3	9.08
t_{10}	-	∅	-	-	-	-	∅	-	2	10.15

Further, to measure the optimal sequence of test cases, a priority is assigned with each test case. The priority can be defined as total faults divide by execution time of each test case. Table 2 illustrates the priority values for each test case. These priorities are computed using table 1. The priority is set to test cases in accordance to decreasing values.

Table 2: Illustrates the priority assigned to different test cases

Test Cases	No. of Faults	Execution Time	Priority
t_1	2	9.3	0.22
t_2	3	10.45	0.29
t_3	2	8..25	0.24
t_4	4	9.48	0.42
t_5	3	11.05	0.27
t_6	2	7.28	0.27
t_7	3	8.46	0.35
t_8	5	10.56	0.47
t_9	3	9.08	0.33
t_{10}	2	10.15	0.20

Hence, the prioritized order computed for ABC prioritization algorithm is highlighted as below.

$T = \{t_8, t_4, t, t_9, t_2, t_6, t_5, t_3, t_1, t_{10}\}$.

Now, APFD is computed for each approach using following equation

$$APFD(T,P) = \langle 1 - \frac{\sum_1^f reval(i,T)}{nf} \rangle + \frac{1}{2n} \quad . \qquad (7)$$

So, the APFD value for ABC prioritized test cases is

ABC Prioritized Algorithm (APFD)=

$$\left(\left(1 - \frac{(5+2+3+4)}{80}\right) + \frac{1}{20}\right) = 0.87$$

The original ordering of test cases is $T = \{t_1, t_2, t_3, t_4, t_5, t_6, t_7, t_8, t_9, t_{10}\}$and APFD value computed for original ordering is

Original Ordering (APFD) =

$$\left(\left(1 - \frac{(2+4+3+4+5+7)}{80}\right) + \frac{1}{20}\right) = 0.73$$

The random ordering of test cases is $T = \{t_2, t_4, t_6, t_9, t_3, t_{10}, t_7, t_1, t_8, t_5\}$ and APFD value for random is

Random (APFD) = $\left(\left(1 - \frac{(3+2+3++10+6)}{80}\right) + \frac{1}{20}\right)$ = 0.75

Reverse ordering cane be described as $T = \{t_{10}, t_9, t_8, t_7, t_6, t_5, t_4, t_3, t_2, t_1\}$ and APFD value for reverse ordering is

Random (APFD) = $\left(\left(1 - \frac{(2+6+6+4)}{80}\right) + \frac{1}{20}\right)$ = 0.82

It is observed that ABC prioritization technique achieves maximum APFD value i.e. 87 as compared to random, reverse and original ordering of test cases. Further, the original ordering of test cases obtains minimum values. So, it can be said that proposed ABC prioritization algorithm provides optimum ordering of test cases.

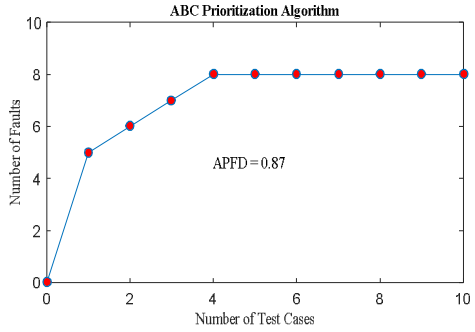

Figure 1: APFD graph using proposed ABC prioritization algorithm

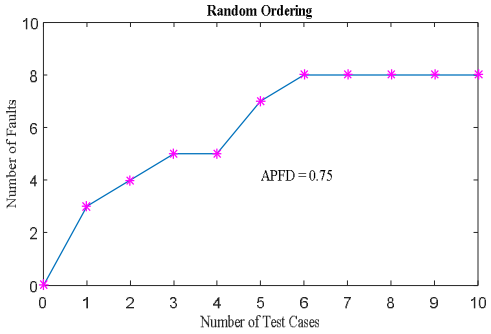

Figure 2: APFD graph using random ordering

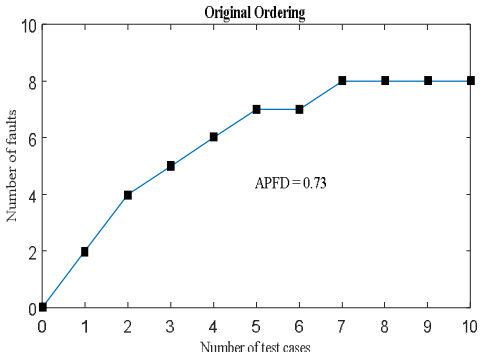

Figure 3: APFD graph using original ordering

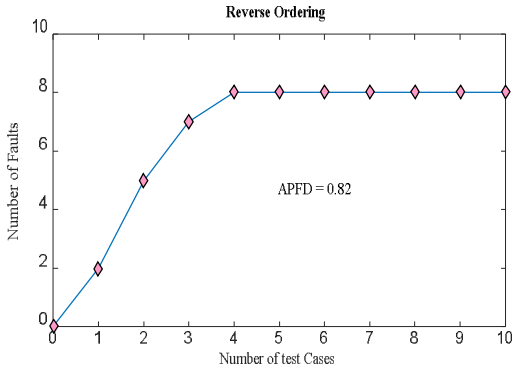

Figure 4: APFD graph using reverse ordering

The comparison between faults detection of proposed ABC prioritization algorithm, randomly prioritized, reverse ordering and original sequence are illustrated in Figures 1-4. Further, the APFD values are used to demonstrate the code coverage. It is seen that the proposed ABC prioritized algorithm having more code coverage than random, reverse and original sequence of test cases. Table 3 shows the APFD values and ranking of ABC prioritized algorithm, random, reverse and original orderingof test cases. It is revealed that the proposed ABC prioritized algorithm obtains first rank; whereas, original sequence of test cases obtains worst rank i.e. 4.

Table 3: APFD values and rank of prioritization algorithms

Parameters	Prioritization Algorithms			
	ABC	Random	Reverse	Original
APFD	0.87	0.75	0.82	0.73
RANK	1	3	2	4

The results of ABC prioritization techniqueis also compared with several other meta-heuristic prioritization algorithms. These prioritization algorithms are GA, PSO and K-means approaches. Table 4 presents simulation results of ABC based prioritization algorithm and other algorithms in terms of APFD metric. It is seen that proposed ABC based prioritization algorithm achieves more accurate results. Hence, it is stated that proposed algorithm efficiently prioritized the test cases for regression testing.

Table 4: APFD values and rank of prioritization algorithms

Parameters	Prioritization Algorithms			
	ABC	GA	PSO	K-Means
APFD	0.87	0.78	0.84	0.76
RANK	1	3	2	4

Further, results of ABC based prioritization algorithm is also tested on several sorting algorithms. These algorithms are merge sort, selection sort, quick sort, heap

sort and insertion sort. The simulation results of proposed algorithm are illustrated in Table 5. Each sorting algorithm is seeded with several faults. Moreover, test cases are also designed for each sorting algorithm. The APFD values and rank of ABC based prioritization algorithm and other prioritization algorithms are mentioned in Table 6. It is observed that ABC prioritization algorithm also provides better results for sorting programs as compared to other prioritization algorithms.

Table 5: Simulation results of ABC algorithm with sorting algorithm

Program Name	Number of Test Cases	No. of Faults	Avg. Execution Time	Optimal Sequence
Merge sort	9	4	16.48	$t_8, t_4, t_2, t_6, t_3, t_7$
Selection Sort	5	3	5.32	t_2, t_3, t_1, t_5
Quick Sort	8	5	13..52	$t_5, t_1, t_7, t_4, t_2, t_8,$
Heap sort	10	3	11.26	$t_4, t_2, t_1, t_5, t_8, t_9, t_3$
Insertion sort	4	4	5.08	t_1, t_3, t_4, t_2

Table 6: APFD values and rank of prioritization algorithms for sorting programs

Parameters	Prioritization Algorithms			
	ABC	GA	PSO	K-Means
APFD	0.83	0.74	0.79	0.72
RANK	1	3	2	4

6. CONCLUSION

This paper presents an ABC prioritization algorithm to prioritize the test cases. To overcome the performance issues of ABC algorithm, some modifications are proposed in original ABC algorithm. The pbest and gbest measures are included in ABC algorithm to improve its convergence rate. The improved ABC algorithm is applied to solve the test suite prioritization problem. The aim of the ABC prioritization algorithm is to compute optimum order of test cases for reducing cost and time of regression testing. The results of ABC prioritization algorithm are compared with random, reverse and original sequence of test cases. The APFD and rank parameters are used to evaluate the performance of abovementioned approaches. It is observed that proposed ABC prioritization algorithm obtains maximum APFD value and having first rank. It is stated that ABC prioritization can reduce the cost and time effectively.

REFERENCES

[1] Beizerm B. Software Testing Techniques, Van Nostrand Reinhold. New York, 1990.

[2] Rothermel G, Untch RH, Chu C, Harrold MJ. Test case prioritization: An empirical study. In: Proceedings IEEE international conference on software maintenance, (ICSM'99),pp 179–188, 1999.

[3] Do H, Rothermel G. On the use of mutation faults in empirical assessments of test case prioritization techniques. IEEE Transaction of Software Engineering, Vol. 32, pp. 733–752, 2006.

[4] Elbaum S., Malishevsky A.G., Rothermel G. Test case prioritization: A family of empirical studies. IEEE Trans SoftwEng, Vol. 28, pp. 159–182, 2002.

[5] Glover F. and Kochenberger G. Handbook of Meta Heuristics, Springer, Berlin, Germany.

[6] Srivastava P., Vijay A., Barukha B., and Sengar P., Sharma R. An Optimized Technique for Test Case Generation and Prioritization Using Tabu Search and Data Clustering," in Proceedings of the 4th Indian International Conference on Artificial Intelligence, pp. pp. 30-46,2009.

[7] Li Z., Harman M., and Hierons R. Search Algorithms for Regression Test Case Prioritization," IEEE Transaction on Software Engineering, vol. 33, no. 4, pp. 225-237, 2007.

[8] Bajwa, J. K., & Kaur, R. An Adaptive Approach For Test Case Prioritization In Regression Testing Using Improved Genetic Algorithm, 2017.

[9] Gladston, A., Nehemiah, K., Narayanasamy, P., & Kannan, A. Test case prioritization for regression testing using immune operator. The International Arab Journal of Information Technology, Vol. 13, No. 6, pp. 686-692, 2016.

[10] Maheswari, R. U., & Mala, D. J. Combined genetic and simulated annealing approach for test case prioritization. Indian Journal of Science and Technology, Vol. 8, No. 35, 2015.

[11] Tulasiraman, M., & Kalimuthu, V. (2018). Cost Cognizant history based prioritization of test case for regression testing using immune algorithm. Journal of Intelligent Engineering Systems, Vol. 11, No. 1, pp. 221-228, 2018.

[12] Bian, Y., Li, Z., Zhao, R., & Gong, D. Epistasis based aco for regression test case prioritization. IEEE Transactions on Emerging Topics in Computational Intelligence, Vol. 1, No. 3, pp. 213-223, 2017.

[13] Chen, J., Zhu, L., Chen, T. Y., Towey, D., Kuo, F. C., Huang, R., & Guo, Y. (2018). Test case prioritization for object-oriented software: An adaptive random sequence approach based on clustering. Journal of Systems and Software, Vol. 135, pp. 107-125, 2018.

[14] Tahvili, S., Afzal, W., Saadatmand, M., Bohlin, M., Sundmark, D., & Larsson, S. Towards earlier fault detection by value-driven prioritization of test cases using fuzzy TOPSIS. In Information Technology: New Generations (pp. 745-759). Springer, Cham, 2016.

[15] Hettiarachchi, C., Do, H., & Choi, B. Risk-based test case prioritization using a fuzzy expert system. Information and Software Technology, Vol. 69, pp. 1-15, 2016.

[16] Noguchi, T., Washizaki, H., Fukazawa, Y., Sato, A., & Ota, K. History-based test case prioritization for black box testing using ant colony optimization. In 2015 IEEE 8th International Conference on Software Testing, Verification and Validation (ICST) pp. 1-2, April 2015.

[17] Jiang, B., & Chan, W. K. Input-based adaptive randomized test case prioritization: A local beam search approach. Journal of Systems and Software, Vol. 105, pp. 91-106, 2015.

[18] Konsaard, P., & Ramingwong, L. Total coverage based regression test case prioritization using genetic algorithm. In 12th international conference on Electrical engineering/electronics, computer, telecommunications and information technology (ECTI-CON), pp. 1-6, June 2015..

[19] Mei, L., Cai, Y., Jia, C., Jiang, B., Chan, W. K., Zhang, Z., &Tse, T. H. A subsumption hierarchy of test case prioritization for composite services. IEEE Transactions on Services Computing, Vol. 8, No. 5, pp. 658-673, 2015.
[20] Panichella, A., Oliveto, R., Di Penta, M., & De Lucia, A. Improving multi-objective test case selection by injecting diversity in genetic algorithms. IEEE Transactions on Software Engineering, Vol. 41, No. 4, pp. 358-383, 2015.
[21] Panwar, D., Tomar, P., & Singh, V. Hybridization of Cuckoo-ACO algorithm for test case prioritization. Journal of Statistics and Management Systems, Vol. 21, No. 4, pp. 539-546, 2018.
[22] Zhao, X., Wang, Z., Fan, X., & Wang, Z. A Clustering-Bayesian network based approach for test case prioritization. In IEEE 39th Annual Conference on Computer Software and Applications (COMPSAC), Vol. 3, pp. 542-547, July 2015.
[23] Tulasiraman, M., Vivekanandan, N., & Kalimuthu, V. Multi-objective Test Case Prioritization Using Improved Pareto-Optimal Clonal Selection Algorithm. 3D Research, Vol. 9, No. 3, pp. 1-13, 2018.
[24] Suri, B., & Singhal, S. Understanding the effect of time-constraint bounded novel technique for regression test selection and prioritization. International Journal of System Assurance Engineering and Management, Vol. 6, No. 1, pp. 71-77, 2015.
[25] Mann, M., Tomar, P., & Sangwan, O. P. Bio-inspired metaheuristics: evolving and prioritizing software test data. Applied Intelligence, Vol. 48, No. 3, pp. 687-702, 2018.
[26] Parejo, J. A., Sánchez, A. B., Segura, S., Ruiz-Cortés, A., Lopez-Herrejon, R. E., &Egyed, A. Multi-objective test case prioritization in highly configurable systems: A case study. Journal of Systems and Software, Vol. 122, pp. 287-310, 2016.
[27] Schwartz, A., & Do, H. Cost-effective regression testing through Adaptive Test Prioritization strategies. Journal of Systems and Software, 115, 61-81.
[28] Marchetto, A., Islam, M. M., Asghar, W., Susi, A., &Scanniello, G. A multi-objective technique to prioritize test cases. IEEE Transactions on Software Engineering, Vol. 42, No. 10, pp. 918-940, 2016.
[29] Karaboga, D., & Basturk, B. A powerful and efficient algorithm for numerical function optimization: artificial bee colony (ABC) algorithm. Journal of global optimization, Vol. 39, No. 3, pp. 459-471, 2007.
[30] Gao W.F., Liu S.Y., Huang L. L. A novel artificial bee colony algorithm based on modified search equation and orthogonal learning. IEEE Trans Syst Man Cybern Part B Vol. 43, pp. 1011–1024, 2013.
[31] Kumar Y. and Sahoo, G. A two-step artificial bee colony algorithm for clustering. Neural Computing and Applications, Vol. 28, No. 3, pp. 537-551, 2017.