International Journal of Engineering and Advanced Technology (IJEAT)
ISSN: 2249-8958 (Online), Volume-9 Issue-1, October, 2019

Flower Pollination Algorithm for Effective Test Case Optimization in Software Testing

Siva Suryanarayana Ch, Satya Prakash Singh

Abstract*: Software testing is measured as an significant way to guarantee software reliability and trustworthiness. Test case optimization shows a major role in software testing and quite a few methods are proposed to improve the fitness of the test case. The effect required to build the software is termed as cost. The fitness value is evaluated over the software that is later considered as cost value and an increase in the fitness value decrease the cost of the software. In this research, the Flower Pollination Algorithm (FPA) is proposed for the test case optimization. FPA is the recently developed algorithm and this is developed based on the function of the flower pollination. The local and global optimization is processed in the ATM machine software and a test case is optimized in the software. The ATM machine function is represented in State chart diagram graph and Sequence diagram graph. Then these two graphs are combined for the effective representation of the withdrawal process. The FPA algorithm has better convergence characteristics than the other Meta heuristic algorithm. The proposed FPA algorithm has a higher fitness value for the test case data is 52.06 %, while existing method has a higher fitness value for the test case data is 45 %.

Keywords*: Flower Pollination Algorithm, Software testing, Sequence diagram graph, State chart diagram graph, and Test case optimization.

I. INTRODUCTION

There are almost more than $500 billion is lost due to the deprived software quality, to solve this problem software testing is carried out. The black-box testing is common software testing method, any enhancement in the increase the software quality have an impact on this loss. Random Testing (RT) and Adaptive Random Testing (ART) are the methods in the black-box, where the test cases are generated without the knowledge of the source code [1]. The specific sampling mechanism is used to scientifically select the test case in the test design methods. The test cases are optimized by these methods to fix the test suite that helps to reduce the cost and time of the software development [2]. Many methods are developed for the software testing and their main aim is to minimize the testing time by analyzing more bugs [3]. Mathematical methods show a considerable analysis in the arrays in some cases. For examples, the orthogonal arrays applied in the design of experiments have the covering arrays in the cases have lower test case [4]. The traditional methods follow a meta-heuristic algorithm that has been executed multiple times to cover all the targets and every iteration with a different target that is processed until all targets are covered [5]. The traditional methods like symbolic execution are finding difficulties in the generation of suitable test input due to the rapid growth of size and complexity of the software [6]. Recently, the search-based software testing (SBST) have much attention and extensive popularity in the software test community. The core idea of the SBST is to use the meta-heuristic algorithms to generate the test data with high coverage ratio and find more bugs in the software [7-10]. In this study, the FPA algorithm is applied to increase the efficiency of the test case optimization. The FPA algorithm has better convergence property compared to the other existing method. The local and global optimizations are carried out in this method and the L'evy flight algorithm is used for representing the insect. The fitness value is inversely proposed to cost value and an increase in the fitness value decreases the cost value. The proposed FPA method is analyzed to understand the performance and this is compared with the existing method. The paper is organized as follows, section II contains a literature survey, the proposed methodology described in section III, the experimental result presented in section IV. The conclusion of this paper made in section 5.

Revised Manuscript Received on October 30, 2019.
* Correspondence Author
Siva Suryanarayana Ch*,Computer Science and Engineering Department, Birla Institute of Technology, Mesra, Ranchi, off campus, Noida, India. cssuryanarayana@bitmesra.ac.in
Satya Prakash Singh, Computer Science and Engineering Department, Birla Institute of Technology, Mesra, Ranchi, off campus, Noida, India. sp.singh@bitmesra.ac.in

II. LITERATURE SURVEY

Bahman Keshanchi, et al. [11] developed a method based on the evolutionary genetic algorithms and heuristic method. The proposed algorithm is evaluated using a behavioral modeling method based on model checking techniques. The proposed method's expected specifications are extracted in the manner of Linear Temporal Logic (LTL) formulas. The cloud model is developed in this method that is fully interconnected with high-speed network. The cross over operator plays a vital role in the Genetic Algorithm (GA) that is used to provide variation in the chromosome of the population. The proposed method significantly decreases the run time than the state-of-art method while the N-GA obtain the close make span of the state-of-art method. The computation time of the method is needed to minimize and the efficiency of the method in the cloud environment has to increase.The evolutionary algorithms are applied in the many cases to identify the near-optimal test suite for software systems. Aldeida Aleti, et al. [12] measured the negative slope coefficient, information content and the number of developments in the process of genetic algorithm in the EvoSuite architecture. The branch, the fitness function and the method coverage are analyzed to find the problem statement in the Evo Suite to revising the objective function. The solution provided by the crossover operator undergoes the mutation operators.

The 19 open source libraries and the program depicted are used to evaluate the performance of the method. The improvement is made continuously instead of the improvements are flanked by stagnation and the better results can be achieved. This shows that the assumption that has better optima are identified at the end of longer gradients. The stopping criteria is one of the important parameter is need to be analysis in this method. Federica Sarro, et al. [13] developed a decision support model with multi-objective to maintain the trade-off between the project risk and time for investigation, so that the software engineers can plan better. This method tested on the six real-world software projects and analyzed with the state-of-art methods and currently used overtime strategies. The proposed method was analyzed, shows the high outcome in all the benchmarks. The effectiveness of the proposed method is needed to be improved in terms of accuracy. L.S. Souza, et al., [14], proposed two multi-objective optimizations (MOO) algorithm like (i) the Binary Multi-Objective Particle Swarm Optimization with Crowding Distance and Roullete Wheel (BMOPSO-CDR) and (ii) a hybrid version with Harmony Search (HS) algorithm (BMOPSO-CDRHS). These two methods are used to solve the issues of the test case with multiple criteria. The developed methods are evaluated in the two different environments: structural testing and functional testing. The output of the method shows that this method has low execution cost and high performance. The hybrid method was evaluated only on the smaller number of programs. Rajesh Ku. Sahoo and Santosh Kumar Nanda [15] proposed a methodology for generation of test cases for an ATM machine withdrawal system using hybrid technique of diagram graph which integrates the state chart and sequence diagram graph (SCSEDG). The Particle Swarm Bee Colony Algorithm (PSBCA) generate the maximum of path coverage. This method is evaluated the effectiveness and efficiency for generating the test cases and to improve the performance. The hybrid method was created based on the Particle Swarm Optimization Algorithm with Bee Colony Algorithm. The proposed PSBCA method is compared with other existing method to handle more test case compared to the other two methods. The test case optimization efficiency is required to improve the execution time. To overcome the above-mentioned issues, the FPA is proposed in this method for the optimization of the test suite in the software.

III. PROPOSED METHOD

In this study, the FPA is used to optimize the test case of the ATM software. The flower pollination is developed based on the process of planting, which has better convergence characteristics than the other metaheuristic algorithm. This algorithm provides the optimal allocation for the test case in the software and then the program is investigated. The graph is designed based on the hybridization of the state chart diagram graph and sequence diagram graph. These graphs are given to the optimization algorithm and the output is investigated. The architecture of the proposed FPA method in software testing is given in Fig (1).

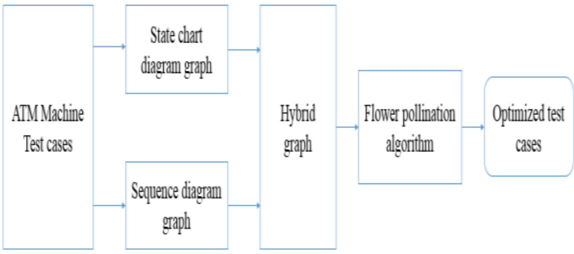

Figure 1. Block diagram of the proposed method

A. Hybridization of State chart Diagram Graph and Sequence Diagram Graph

ATM machine software is used to analyze the performance of the proposed FPA method in the software testing and dependency of the ATM software function is also explained in the research [15]. The function of the ATM is converted into the graph for effective representation. The graph is input to the optimization process of flower pollination. The ATM function is represented in the SCDG, SEDG and integrated for analyzing function [15]. The integration of the graphs are depends on the pseudo code, given in table 2. The integration of these two charts provide the hybrid chart graph [15], is shown in Fig. (1).

Input: - State chart Diagram Graph (SCDG) and Sequence
Diagram Graph (SEDG)

Output: - State chart Sequence Diagram Graph (SCSEDG)

1. P= analyze all possible ways of SCDG
2. For each path p_i, $\in P$ do
3. $S_j = S_i$ // Current state S_j is initialized with initial state S_i
4. $T \leftarrow \phi$ // T is initialized with a NULL value
5. For each state S_i of the path p_i

If $ci \in Si$ // Check if current state Si have multiple derived paths

$\alpha = Si - 1 \rightarrow SEDG$

$\beta = SEDG\ (last) \rightarrow Si + 1$

End If

$T \leftarrow T \cup T1$

If $ci \notin Si$

$\gamma = Si \rightarrow Si + 1$

End If

6. End For
7. End

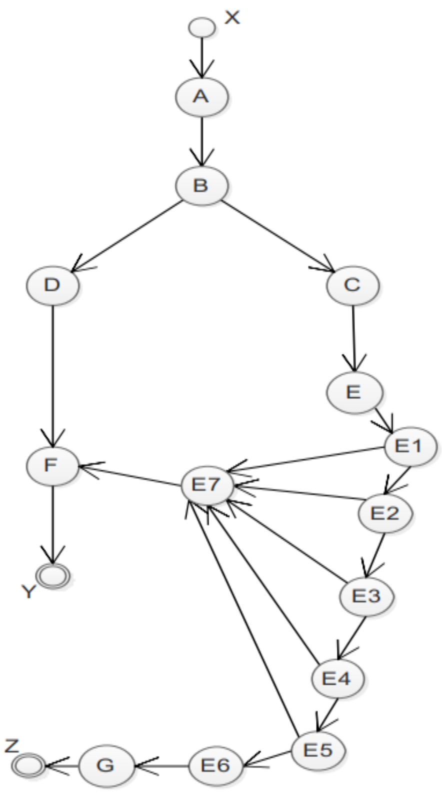

Figure 2 State chart sequence diagram graph of withdrawal operation of ATM

The starting node is X, the Y and Z are the end nodes. The unsuccessful process ends in Y node and the successful process ends in Z node. The intermediate nodes such as A, B, C, D, E, E1, E2, E3, E4, E5, E6, E7, F and G denotes the various process or activities compute within the system during the execution. The intermediate node between A-G process the withdrawal function and the node E1-E7 perform the withdrawal amount checking operation. These graphs are given to the FPA and using these test case, the proposed method provides the optimized program.

B. Flower pollination Algorithm for the test case Optimization

The FPA is a recent metaheuristic algorithm, which is developed by the Yang [16]. This algorithm is developed based on the planting process of the flower and this will find the optimized value in the pollination behavior and flower constancy are considered. This method is applied for the different application [17, 18, and 19] such as sizing and global optimization. Yang [16] described the FPA in the four steps of pollination incorporating flower constancy and the behavior of pollination. The four important steps are explained as follows.

The cross-pollination and Biotic have considered as the global pollination with the pollen contain pollinators. This is processed based on the L'evy flights method.

Self-pollination and abiotic process are the local pollinations.

The flower constancy is denoted as the reproduction probability that is proportional to the similarity of the two flowers involved.

The switch probability $p \in [0,1]$ is used to control the local and global pollination. The physical proximity and the other aspects such as local pollination, and wind have a significant fraction p in the overall activity of the pollination.

Each plant have the several flowers and patch of each flower releases the millions of pollen gametes. To simplify the terms, the FPA method are consider that each plant has one flower and each flower will produce only one pollen gamete. The differentiation is not required for the pollen gamete, a plant, and solution to a problem or a flower. For the multi-objective optimization problems, the pollen gametes can be easily extended for each flower to increase the performance. Based on this function, this flower depended optimization method is named as the FPA. The two major process in this algorithm namely global pollination and local pollination. In the global pollination step, the insects are set for the flower pollens as a pollinator, and the pollen can carry out the long places because insert can fly to the long distance places. This ensures the reproduction and fittest pollination, and thus represent the fittest as$g *$. The first rule and the flower constancy is denoted mathematically in the Eq. (1).

$$X_i^{t+1} = X_i^t + L(X_i^t - g_*), \qquad (1)$$

In x_t^i, the pollen is denoted as i or solution vector x_i at iteration t and the current best solution is represented by $g *$ is identified from all solutions at the iteration. The pollination strength is represent as L, which is important in a step size. The insect move over the long distance, so flight technique is mimic to characteristic effectively. The Levy distribution is represented in the Eq. (2) with $L > 0$.

$$L \sim \frac{\lambda\Gamma(\lambda)\sin(\pi\lambda \backslash 2)}{\pi}\frac{1}{s^{1+\lambda}}, \quad (s \gg s_0 > 0) \qquad (2)$$

The standard gamma function $\Gamma(\lambda)$ is valid for large steps $S > 0$. The gamma function is set as 1.5 through the experiment. Then the local pollination or the Rule 2 involves in the flower constancy is denoted in the Eq. (3).

$$X_i^{t+1} = x_i^t + \epsilon\,(X_j^t - X_k^t), \qquad (3)$$

Where x_t^j and x_t^k are the pollens of the different flowers present in the same plant and this is regarded as the flower constancy in a neighborhood. Selecting these two pollens from the same population involves in causing a local random walk if that is plots ϱ from uniform distribution in [0, 1]. Most flower pollination activities are carried out in the global and local scale. The adjacent flower patch or the flower that are present near are pollinate locally than those are present in far. The switching probability in the Step 4 or proximity probability p is change between the global and local pollination. The initial value of the algorithm is set as $p = 0.5$ in the value and then investigate the parametric study in the most appropriate range.

Pseudo Code for Flower pollination algorithm

Objective function minimum or maximum $f(x), x = (x_1, x_2, \ldots, x_d)$

Population initialization based on n flowers/pollen gametes and set with random solutions

The best solution $g*$ is identified from the initial population

Switch probability assignment $p \in [0, 1]$

while ($t < MaxGeneration$)

for $i = 1 : n$ (all n flowers population)

if $rand < p$,

Plot a step vector L that follows a L´evy method

Global pollination through $x_t^i + 1 = x_t^i + L(g* - xt\ i)$

Else

Plot ϱ from a uniform distribution in [0,1]

Choose j and k randomly in all solutions

Perform local pollination through $x_t^i + 1 = x_t^i + \varrho(x_t^j - x_t^k)$

end if

Measure the new solutions

In case new solutions are better, replace in the population

end for

Identify the current best solution $g*$

end while

IV. EXPERIMENTAL RESULT

The test suite optimization was carried out by the FPA and this is evaluated by the ATM program. The FPA has recently developed algorithm and this is applied to the different application like sizing and global optimization. The evaluated results of the test case optimization using FPA are investigated in this section. The test case optimization are simulated in the scenario of 4 GB RAM in the Intel Duo core processor of 500 GB hard disk. The process was carried out in the tool of Netbeans. The test case optimization is processed for the different method for the same software environment and fitness value is measured. The existing and proposed method is investigated in the performance and compared with each other in Table 1.

The major objective of this method is to find the best fitness value for the test case in the software. The FPA method is compared with other state-of-art methods such as PSA, BCA and PSBCA. The fitness value is measured for the different number of iterations and compared with other methods. The table 1 shows that the proposed FPA method has a higher fitness value than the other methods. The FPA algorithm process 44000 test cases in the 30 iterations while state-of-art methods requires 120 iterations to process same test case. This shows the efficiency of the proposed FPA method over another existing method. So, the proposed method has the capacity to handle a large number of data in less iteration. The graphical representation of a test case with the various iteration is shown in Fig (3).

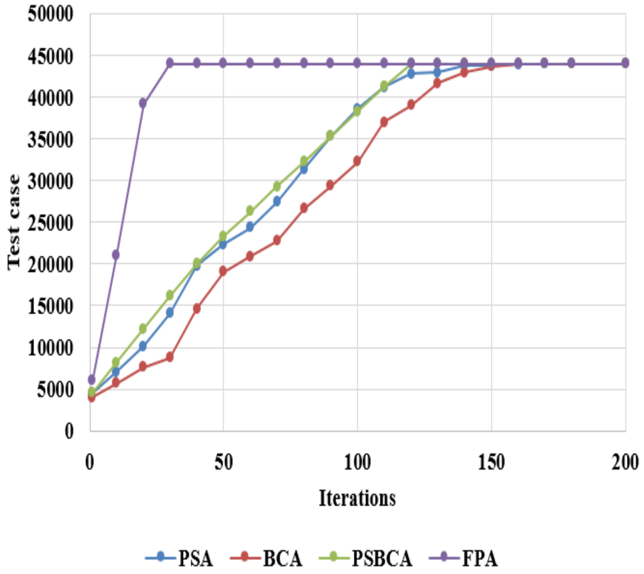

Figure 3 Test case in the various iterations

In the first iteration, the proposed FPA method process 6000 test case while state-of-art method process 4600 test case. The proposed FPA method is compared with three existing methods and the FPA has more test functions in the fewer iterations. Fig (3) shows the efficiency of the proposed method over the other three existing methods. The FPA method process the 44000 test data in the 30 iterations and other state-of-art method requires more than 100 iterations to process the 44000 test data. The test case fitness value in terms of test data percentage is shown in Table 3.

Table 1 Test case with the maximum fitness value

FITNESS VALUE RANGE	% OF TEST CASES /TEST DATA (PSA)	% OF TEST CASES /TEST DATA (BCA)	% OF TEST CASES /TEST DATA (PSBCA)	% OF TEST CASES /TEST DATA (FPA)
0 ≤ f(x) < 0.3	45	40	30	13.6
0.3≤ f(x) < 0.7	30	35	25	34.34
0.7≤ f(x) < 1.0	25	25	45	52.06

The test case fitness values are categorizing into three range as: 0-0.3, 0.3-0.7 and 0.7-1.0. The first category represents low fitness value than the other two categories and third category has the higher fitness value of test case. The FPA method has the 52.06% of the test case data which are presented in the third category, while the state-of-art method has 45% of the test case in the third category. The proposed method has 13.6 % of the test data in the first category and the state-of-art method has 13.6 % of the data in the first category.

Table 2 FPA and existing method in software testing

	Particle Swarm Algorithm(PSA)		Bee Colony Algorithm(BCA)		Particle Swarm Bee Colony Algorithm (PSBCA) [15]		Flower Pollination Algorithm	
Iteration Number	Test cases /Test data	Fitness Function Value	Test cases /Test data	Fitness Function Value	Test cases /Test data	Fitness Function Value	Test cases/Test data	Fitness Function Value
1	4500	6.10E-10	4000	5.95E-10	4600	6.13E-10	6000	7.34E-07
10	7100	6.96E-10	5700	6.47E-10	8200	7.38E-10	21110	3.34E-06
30	14100	1.05E-09	8800	7.63E-10	16200	1.21E-09	44000	0.0013
50	22400	1.96E-09	19100	1.49E-09	23300	2.12E-09	44000	0.00167
60	24400	2.36E-09	20900	1.72E-09	26300	2.86E-09	44000	0.00194
70	27400	3.23E-09	22800	2.03E-09	29300	4.06E-09	44000	0.00215
100	38600	2.44E-08	32300	6.20E-09	38300	2.23E-08	44000	0.00277
110	41200	6.92E-08	37000	1.56E-08	41300	7.30E-08	44000	0.00303
120	42800	2.07E-07	39000	2.78E-08	44000	1.00E-06	44000	0.00346
170	44000	1.00E-06	44000	1.00E-06	44000	1.00E-06	44000	0.06452
180	44000	1.00E-06	44000	1.00E-06	44000	1.00E-06	44000	0.06098
200	44000	1.00E-06	44000	1.00E-06	44000	1.00E-06	44000	0.11044

This shows that the proposed FPA method has a higher percentage of test case than the other three existing methods. The comparison of the different method in the fitness value in the different range is shown in Fig (4).

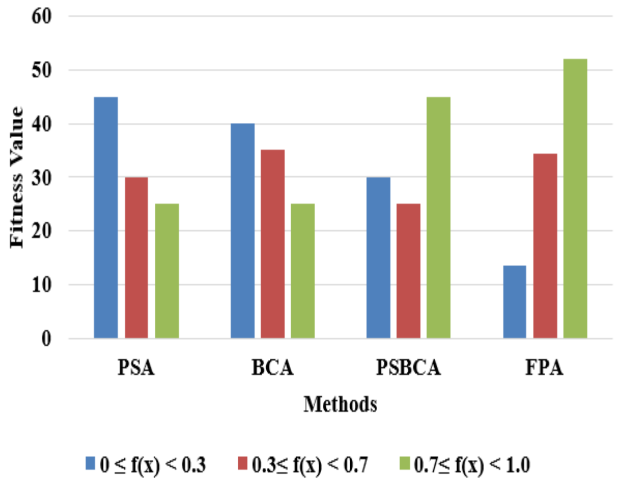

Figure 4 Fitness value of several methods

The fitness value present in the range of the percentage of the test data for the various method is compared in the graphical representation, as shown in Fig (4). The proposed method has a higher fitness value for more test case than the existing method. The PSBCA has considerable fitness value compared to the PSA and BCA method in software testing. The proposed FPA technique has the higher efficiency than the other state-of-art methods in software testing.

Hence, the proposed FPA has higher performance than the other state-of-art method. The experimental results show the proposed FPA has the capacity to handle the more test cases in the less number of iterations.

V. CONCLUSION

Software testing is a difficult process due to a large number of test case present in the software and automatic test case optimization process can be applied to improve the efficiency. FPA is proposed in this study to increase the efficiency of the test case optimization. The FPA is chosen for its better convergence properties than the other state-of-art method. The bank ATM machine program is chosen for the validation of the proposed FPA method. The ATM withdrawal process is represented in the two graphs and these are combined for the effective representation. This graph is given to the FPA and optimization is carried out by the flower pollination function. The global and local optimization is carried out by the FPA methods in the test case. The L'evy flight is simulated for the insect to carry the flower pollination in the system. The result of the proposed method has a higher fitness value than the state-of-art method. The FPA method provides the highest fitness value for 52.06 % of testing data while the existing method has the 45 % testing data for higher fitness value. The more fitness value is processed with less number of iteration is highly denote the cost value of software. Although, the proposed method processes the number of data in the less iteration than the other methods. In future work, the FPA method will be in the software in the Service Oriented Architecture.

REFERENCES

1. Shahbazi, A. and Miller, J., 2016. Black-box string test case generation through a multi-objective optimization. IEEE Transactions on Software Engineering, 42(4), pp.361-378.
2. Ahmed, B.S., Abdulsamad, T.S. and 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, pp.13-29.
3. Mann, M., Sangwan, O.P. and Tomar, P., 2015. Hybrid test language processing based framework for test case optimization. CSI transactions on ICT, 3(2-4), pp.71-81.

4. Panichella, A., Kifetew, F.M. and Tonella, P., 2018. Automated test case generation as a many-objective optimisation problem with dynamic selection of the targets. IEEE Transactions on Software Engineering, 44(2), pp.122-158.
5. Wu, H., Nie, C., Kuo, F.C., Leung, H. and Colbourn, C.J., 2015. A discrete particle swarm optimization for covering array generation. IEEE Transactions on Evolutionary Computation, 19(4), pp.575-591.
6. Mao, C., Xiao, L., Yu, X. and Chen, J., 2015. Adapting ant colony optimization to generate test data for software structural testing. Swarm and Evolutionary Computation, 20, pp.23-36.
7. Khari, M., Kumar, P., Burgos, D. and Crespo, R.G., 2018. Optimized test suites for automated testing using different optimization techniques. Soft Computing, pp.1-12.
8. Kessentini, W., Kessentini, M., Sahraoui, H., Bechikh, S. and Ouni, A., 2014. A cooperative parallel search-based software engineering approach for code-smells detection. IEEE Transactions on Software Engineering, 40(9), pp.841-861.
9. Mann, M., Tomar, P. and Sangwan, O.P., 2018. Test Data Generation Using Optimization Algorithm: An Empirical Evaluation. In Soft Computing: Theories and Applications (pp. 679-686). Springer, Singapore.
10. Ahmed, B.S., Gambardella, L.M., Afzal, W. and Zamli, K.Z., 2017. Handling constraints in combinatorial interaction testing in the presence of multi objective particle swarm and multithreading. Information and software Technology, 86, pp.20-36.
11. Keshanchi, B., Souri, A. and Navimipour, N.J., 2017. An improved genetic algorithm for task scheduling in the cloud environments using the priority queues: formal verification, simulation, and statistical testing. Journal of Systems and Software, 124, pp.1-21.
12. Sarro, F., Ferrucci, F., Harman, M., Manna, A. and Ren, J., 2017. Adaptive multi-objective evolutionary algorithms for overtime planning in software projects. IEEE Transactions on Software Engineering, 43(10), pp.898-917.
13. Aleti, A., Moser, I. and Grunske, L., 2017. Analysing the fitness landscape of search-based software testing problems. Automated Software Engineering, 24(3), pp.603-621.
14. de Souza, L.S., Prudêncio, R.B.C. and de Barros, F.A., 2015. A hybrid particle swarm optimization and harmony search algorithm approach for multi-objective test case selection. Journal of the Brazilian Computer Society, 21(1), p.19.
15. Sahoo, R.K., Nanda, S.K., Mohapatra, D.P. and Patra, M.R., 2017. Model Driven Test Case Optimization of UML Combinational Diagrams Using Hybrid Bee Colony Algorithm. International Journal of Intelligent Systems and Applications, 9(6), p.43.
16. Yang, X.S., 2012, September. Flower pollination algorithm for global optimization. In International conference on unconventional computing and natural computation (pp. 240-249). Springer, Berlin, Heidelberg.
17. Bekdaş, G., Nigdeli, S.M. and Yang, X.S., 2015. Sizing optimization of truss structures using flower pollination algorithm. Applied Soft Computing, 37, pp.322-331.
18. Abdelaziz, A.Y., Ali, E.S. and Elazim, S.A., 2016. Combined economic and emission dispatch solution using flower pollination algorithm. International Journal of Electrical Power & Energy Systems, 80, pp.264-274.
19. Cui, W. and He, Y., 2018. Biological Flower Pollination Algorithm with Orthogonal Learning Strategy and Catfish Effect Mechanism for Global Optimization Problems. Mathematical Problems in Engineering, 2018.

AUTHORS PROFILE

Siva Suryanarayana Ch is completed MCA in 1996 from Osmania University, Hyderabad. M.Tech (Information Technology) in 2005 from AAI, Allahabad .I have guided more than 100 Graduate student's projects. Total teaching experience 24 years and 6 papers published in National and International journals.

Satya Prakash Singh is completed M. Sc(Physics) in 1997 from Dr B R Ambedkar University, Agra. M. Tech(Computer Science & Engineering) in 1999 from Kurukshetra University, Kurukshetra and Ph. D in 2012 from JJT University, Jhunjhunu, Rajasthan. Two Ph. D students completed Ph.D. under my supervision. I have guided more than 100 Post Graduate student's projects. Total teaching experience 20 years and more than 40 papers published in National and International journals.