Research Journal of Applied Sciences, Engineering and Technology 8(1): 76-82, 2014
DOI:10.19026/rjaset.8.943
ISSN: 2040-7459; e-ISSN: 2040-7467

Submitted: March 02, 2014 Accepted: April 17, 2014 Published: July 05, 2014

Research Article

Fuzzy C Means (FCM) Clustering Based Hybrid Swarm Intelligence Algorithm for Test Case Optimization

[1]Abraham Kiran Joseph and [2]G. Radhamani
[1]Dr. G.R. Damodaran College of Science,
[2]Department of Computer Science, Dr. G.R. Damodaran College of Science,
Affiliated to Bharathiar University, Tamilnadu, India

Abstract: The main objective of an operative testing strategy is the delivery of a reliable and quality oriented software product to the end user. Testing an application entirely from end to end is a time consuming and laborious process. Exhaustive testing utilizes a good chunk of the resources in a project for meticulous scrutiny to identify even a minor bug. A need to optimize the existing suite is highly recommended, with minimum resources and a shorter time span. To achieve this optimization in testing, a technique based on combining Artificial Bee Colony algorithm (ABC) integrated with Fuzzy C-Means (FCM) and Particle Swarm Optimization (PSO) is described here. The initiation is done with the ABC algorithm that consists of three phases-the employed bee, the onlooker bee and the scout bee phase. The artificial bees that are initialized in the ABC algorithm identify the nodes with the highest coverage. This results in the ABC algorithm generating an optimal number of test-cases, which are sufficient to cover the entire paths within the application. The node with the highest usage by a given test case is determined by the PSO algorithm. Based on the above 'hybrid' optimization approach of ABC and PSO algorithms, a set of test cases that are optimal are obtained by repeated pruning of the original set of test cases. The performance of the proposed method is evaluated and is compared with other optimization techniques to emphasize the fact of improved quality and reduced complexity.

Keywords: Hybrid swarm intelligence algorithm, optimization algorithm, software testing, test cases

INTRODUCTION

An organized testing methodology increases the reliability and the confidence of the end user in a software application (Mahapatra and Singh, 2008). Testing strategy aims at improving the overall quality of the software, by eliminating bugs that range from cosmetic ones to serious ones that are of high priority and severity. The down side to such an extensive testing method could be the time constraint and cost (Binder, 2000). A testing process not only focuses on identification and specification of defects but also offer feasible steps to be taken for removing these defects (Korel, 1990). For any given application, even of medium complexity an exorbitant number of test cases would be necessary. To reduce the effort in testing, test cases that reveal the maximum number of undetected errors could be chosen. Developing an optimized technique to identify only these test cases remains a challenge even now (Clarke, 1976).

The majority of existing test suite reduction approaches could minimize the size of test suites considerably (Zhong *et al*., 2003). The main aim of test case execution is to identify faults in the software; fault detection capability is one of the main measures in this aspect. A drawback that is observed here is the permanent elimination of test cases from a test suite. This may highly decrease the fault detection effectiveness of the remaining test cases in the suite. The tradeoff between time required to execute test cases and their rate of fault detection should be taken into account when applying test suite reduction approaches (Rothermel *et al*., 1998).

Meta-heuristic techniques have been used productively in test case generation and prioritization. Complexity and time consumption for convergence are two issues that are to be addressed during optimization. The approach here uses swarm intelligence based techniques for test case optimization. A number of swarm intelligence approaches have been observed to produce comparatively better results in terms of accuracy, convergence behavior and time duration for overall test execution. In recent years, more research has been done in hybrid optimization techniques and the results suggest they provide better results than individual optimization techniques.

This study uses a couple of recent swarm intelligence methods called as the Particle Swarm

Corresponding Author: Abraham Kiran Joseph, Dr. G.R. Damodaran College of Science, Affiliated to Bharathiar University, Tamilnadu, India

Algorithm and Artificial Bee Colony integrated with Fuzzy C-Means (ABC-FCM) Algorithm for Test Case Optimization (Krishnamoorthi and Natarajan, 2013). The research reinstates that the optimized suite produced by PSABC covers the faults entirely in a program when compared with other approaches.

LITERATURE REVIEW

Software Test related problems may not be solved with a conventional software engineering system. Mathematically designed formulations, by means of meta-heuristics could be used to solve these problems. In Fabrício *et al*. (2010) the author introduces a Search Based Software Engineering (SBSE), to solve the software engineering problem. In recent years, specific subarea called Search Based Software Testing (SBST) has become common. The experimentation results based on modern SBST have provided good results.

The aim here is to generate a set of test cases in such a way that the cost for finding the defects in the software system is reduced considerably. In a test suite, the test sequences are generated in a specific order to attain maximum coverage. In Li and Lam (2011) an Ant Colony Optimization approach (ACO) to automatic test sequence generation is used for state-based software testing. This approach uses UML artifacts to automatically produce test sequences to attain the required test coverage.

Kaur and Goyal (2011) offered the Bee Colony Optimization (BCO) algorithm to determine fault coverage. In the bee colony, there are two types of worker bees, scout bees and forager bees. These bees are applied for the growth and protection of the colony (Al-Tabtabai and Alex, 1999). The fault coverage in a regression test suite is based on the performance of these two bees. The BCO algorithm is considered to get maximum fault coverage, whose results are similar to an optimal solution.

Regression Testing is an inevitable and a very upscale process to be carried out, frequently in a time and resource constrained environment. Test case selection and prioritization are the methods, to choose and prioritize a subset from the total test suite, satisfying several criterions (Bansal, 2013).

Krishnamoorthi and Sahaaya Arul Mary (2009) introduced a test case prioritization method by Genetic Algorithm (GA). This approach prioritizes subsequences of the unique test suite so that the new suite, which is run inside a time-constrained execution situation, will have an improved rate of fault detection when evaluated to rates of arbitrarily prioritized test suites. The experimental result shows that the algorithm is more productive in terms of performance and time constraints. An Average Percentage of Faults Detected (APFD) metric is used to resolve the efficiency of the new test case. In Doungsaard *et al*. (2008) an approach for generating test data from UML state diagram by means of genetic algorithm is explained. This facilitates the reduction of risks in producing test data before coding, to a considerable degree. In Kaur and Goyal (2011) a Genetic Algorithm is used to prioritize the regression test suite. It prioritizes test cases based on whole code coverage. An Average Percentage of Code Covered (APCC) metric is used to show the effectiveness of the algorithm.

Kennedy and Eberhart developed the Particle Swarm Optimization (PSO) Algorithm in 1885. PSO is based on the social behavior of a set of migrating birds trying to land in an unknown destination. Here, each solution is a 'bird' in the flock and is known to as a 'particle'. A particle corresponds to a chromosome or population member in Tabtabai and Alex (1999). Unlike GA, the PSO won't create new birds from parent ones in an evolutionary procedure. As an alternative, the birds in the population only evolve their social behavior and as a result their movement towards a destination gets refined in an incremental way ads discussed by Shi and Eberhart (1998). The process is initiated with a collection of random particles (solutions), N. The i^{th} particle is denoted by its position as a point in an S-dimensional space, where S denotes the number of variables.

In ABC algorithm, the solution of the optimization problem is indicated by the food source position and the quality of the solution is referred to as the nectar amount of the source. In the initial step of ABC, the locations for the food source are produced randomly. In other words, for SN (the number of employed or onlooker bees) solutions, a randomly distributed initial population is produced. In the solution space, each solution ($X_i = (X_{i1}, X_{i2}, \ldots \ldots, X_{iSN})$) is a vector with its number of optimization parameters (Sonmez, 2011).

METHODOLOGY

Artificial Bee Colony integrated with Fuzzy C-Means (ABC-FCM): The FCM function is incorporated with ABC algorithm in the proposed approach. This hybrid algorithm is modelled to gain the positive attributes of the ABC and FCM algorithms. The ABC algorithm consists of three phases namely the employed bee phase, the onlooker bee phase and the scout bee phase. The employed bee phase and onlooker phase are predictable phases while the scout bee phase is generated as a random phase. The FCM operator is incorporated in the scout bee phase of the ABC algorithm.

In ABC algorithm (Dervis and Ozturk, 2011) a random solution is produced for the scout bees, when an abandon solution takes place. This random solution may not be conclusive of producing a reliable solution. The proposed algorithm provides the replacement solution, in each cycle where in the solution for scout bees are presented by the FCM operator. The new solution from the FCM operator is constructed based on

the solutions of the employed bee and onlooker bee phases for better optimization results. The fitness function is calculated using Eq. (1):

$$fit_i = \begin{cases} \frac{1}{1+f_i}, if\ f_i \geq 0 \\ 1 + abs(f_i), if\ f_i \leq 0 \end{cases} \quad (1)$$

The data are initialized in an orderly way, corresponding to the number of clusters. The ABC algorithm is used to define centroids for the processing. The dataset can be measured as the subsequent set with 'n' elements:

$$D = \{d_1, d_2, \ldots d_n\} \quad (2)$$

Assume that the two clusters are created from the given dataset. Two centroids for the clusters are chosen from the dataset D:

$$\begin{matrix} & d_1 & d_2 & \ldots\ldots & d_n \\ c_1 & i_1 & i_2 & \ldots\ldots & i_n \\ c_2 & j_1 & j_2 & \ldots\ldots & j_n \end{matrix}$$

where, c_1 and c_2 represent the two centroids of the two clusters, respectively. i and j be the demonstration of distance values among the centroids and the data points. After the distance calculation, the data are moved into the cluster, which has the least distance value when compared with other distance values of the data point. Finally, the data points are grouped into two clusters according to their least distance value. The f_i values of the clusters are measured from the distance value of the data points. The fitness function is calculated as the sum of all the f_i values Eq. (3):

$$fitness\ function = \sum_{i=0}^{n} f_i \quad (3)$$

Employed bee phase: The ABC algorithm is a multidimensional search space, in which there are employed bees and onlooker bees. These bees are used to find the food sources. The presented data are chosen and their corresponding solutions are arbitrarily produced by means of uniform distribution. The initial population is then chosen for the employed bee phase.

Consider the solution in Table 1, which is the solution generated after the initialization process. Applying Eq. (4), the solution is altered in the employed bees. These employed bees have the food location. The position values are altered as indicated by 'I'.

Onlooker bee phase: E stands for the employed bee variants and O for the onlooker bee variants. In the employed bee phase, the E variants are the solution for the first onlooker bee in Table 2 and it can be calculated by the following formula:

Table 1: Clusters and distances

C1	C2
i_1	j_1
i_2	j_2

Table 2: Sample solution

I_1	I_2
I_5	I_6
I_3	I_4
I_7	I_8

$$V_{i,j} = E_{i,j} + \emptyset_{i,j}(E_{i,j} - E_{k,j}) \quad (4)$$

where, k and j are random indices and $\emptyset_{i,j}$ is a randomly produced number in the range (-1, 1). A new solution is generated by the above equation. The solution deals with the fitness function for getting the fitness value. The new fitness value is compared with the prior best value. Here, the ABC algorithm first starts processing the employed bee in the cycle. The distance values are iteratively changed by altering the indices k and j in the Eq. (2). Thus, a set of data with different fitness values are generated. Among those set of distance values, the set of values with higher fitness value is selected for the scout bee phase.

Scout bee phase: The scout bee is a randomly assigned bee in the ABC algorithm, if an abandon solution occurs. In other words, if there is no new solution obtained at the end of the cycle, a bee position will be randomly assigned to get a new solution. In the proposed approach (Fig. 1), a new method is used to introduce the scout bee, in the case of an abandon solution. Instead of adding a random position to the bee colony, the proposed approach uses the FCM function to introduce the scout bee. The scout bee is produced from the onlookers with the highest fitness, though they don't possess a better fitness value than the previous one. The onlookers with high fitness are sorted in their ascending order of their fitness values:

$$O_i = [p_i, p_2, \ldots\ldots, p_m] \quad (5)$$

Here, O_i is the set of onlooker bee positions, which has the highest fitness. A scout bee is generated from the processing of the above set with the help of the FCM function. Thus, the new scout bee can be generated from the following function Eq. (6) and (7):

$$C_j = \frac{\sum_{i=1}^{n} m_{ij}^{n}.p_i}{\sum_{i=1}^{n} m_{ij}^{n}} \quad (6)$$

$$m_{ij} = \frac{1}{\sum_{i=1}^{c}(|p_i - c_j|)^{\frac{2}{x-1}}} \quad (7)$$

where,

m_{ij} = The fuzzy membership value

C_j = Centriod calculated for the set

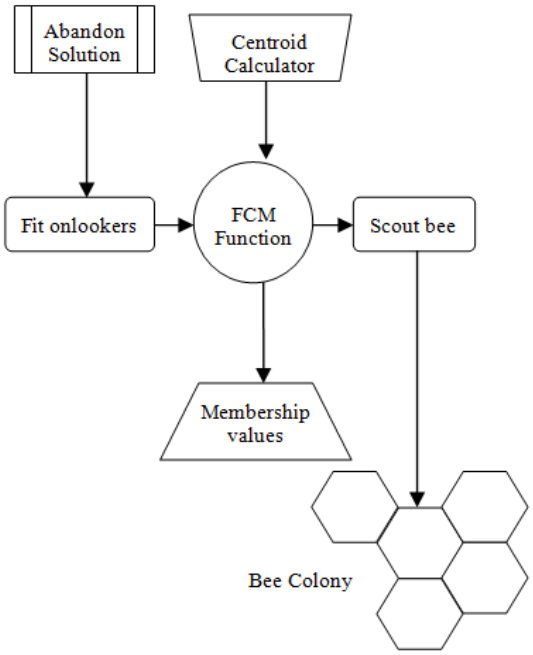

Fig. 1: The scout bee phases

The FCM function generates a new position for the scout from the membership values and the centroid values. As mentioned above, only one centroid is defined since a single scout bee has to be introduced to the context. Unlike the FCM algorithm, in the proposed approach a single iteration is conducted to obtain the new solution. The main advantage in the proposed approach is that the scout bee is added to the solution by processing from the best fitness values obtained from the last solution in the cycle instead of randomly assigning a scout bee. Moreover, the scout bee is introduced in every cycle, which improvises the optimal solution and the overall execution time.

Hybrid form of ABC-FCM-PSO for the test case optimization: In this approach, a combined form of PSO algorithm and ABC-FCM algorithm is used for optimizing the Test Cases. In the proposed methodology each test case would symbolize a food source and the objective of this method is to find a best food source, i.e., it refers to the test cases with maximum coverage in terms of statements and faults. The food source position corresponds to a potential solution of the optimization problem and the nectar amount match to the fitness of the associated solution.

This research study proposes that the optimized test suite produced by the algorithm will cover all possible statements and faults in the program. The program is given to the proposed Test Case optimization approach. The distinction of the proposed algorithm is that it uses a Fuzzy C Means (FCM) operator in the Artificial Bee Colony (ABC) algorithm. The role of the FCM operator begins at the scout bee phase of the ABC algorithm, the scout bees being introduced by the FCM operator. ABC-FCM is functional to produce an Optimal Test suite by generating test data which would have higher statement and path coverage. The test data will be the required input to be given to the System under Test (SUT), for travelling along the path and vice versa.

Initially, the program is given to the Test Case optimization tool, which transforms the corresponding program into an equivalent Control Flow Graph (CFG). The independent paths from the start node to the end node are produced from the CFG. Each independent path consists of a number of normal nodes and predicate nodes. Every independent path would denote a Test Case (Pressman, 2007). ABC algorithm generates an optimal Test suite by traversing through independent test cases. The search bee would be the search agent which looks for the execution state of the SUT and also initiates the test cases with the initial test data through equivalence partitioning and boundary value analysis (Reid, 1997). Then the search agent computes the fitness value of each test node through evaluating the coverage of each node. This is repeated until an executable state of the SUT is determined.

The search bee then gives the fitness value of the traversed nodes/neighbouring nodes to the chosen agent (Srikanth *et al*., 2011). The chosen bee evaluates the fitness value of traversed nodes and the neighbouring nodes. If the fitness value of the node obtained is greater than the neighbouring node's fitness value, the node's information is stored in the optimal test case repository. The node whose fitness value is less is discarded.

The algorithm for test case optimization using ABC-FCM-PSO algorithm approach is seen below. In order to implement any algorithm, the algorithm must be converted into the pseudo code before programmatically developing into an application. The detailed pseudo code of the Test Case Optimization algorithm using ABC-FCM-PSO approach is presented in the following section.

The process is initiated with a collection of random particles (solutions), N. The i^{th} particle is denoted by its position as a point in S-dimensional space, where S denotes the number of variables.

The following is the detailed algorithm:

1. Initialize the test case which is to be performed by the search bee in the algorithm, then search for executable state and evaluate the test cases
2. First of all, initialize the current traversal path
3. Generate the initial population of the employee bees and select the half bee employed with the Particle Swarm Optimization (PSO)
4. Calculate the distance matrix
5. Evaluate the fitness value for the initialized population
6. Repeat the process for the onlooker bee
7. For each onlooker bee calculate the new test cases V_i and find the fitness value for that new solution by applying greedy process

8. Select the bee with Least fit value
9. Apply FCM operator in the selected bee to find scout bee
10. Add scout bee into employed bee
11. Repeat steps 4-11
12. Probability value for the new solution is calculated
13. Above two processes is repeated for the onlooker bee then replace it with the obtained new solution which will be randomly produced and it is stored
14. Add the test case to the optimal repository
15. In the next iteration scout bees generate the new test data

Then ABC-FCM iterates till its stopping criterion is met which is the maximum number of paths and faults covered. Then the optimal values of individuals obtained from the ABC-FCM are given to the PSO as the starting point. Then, PSO produces its first individual sets, in which the starting point is by provided by the initiating point for the PSO algorithm is produced by the ABC-FCM.

To implement the above algorithm, the proposed approach uses the Test Suite Optimization tool to optimize the Test Cases by using the ABC-FCM-PSO algorithm. The tool considers a program as an input to generate independent paths. With the generated independent paths, Test Cases are traversed along the paths with the help of ABC-FCM-PSO algorithm. The test cases with maximum coverage are identified as a result of this. At last the optimal Test Suite is produced as an output.

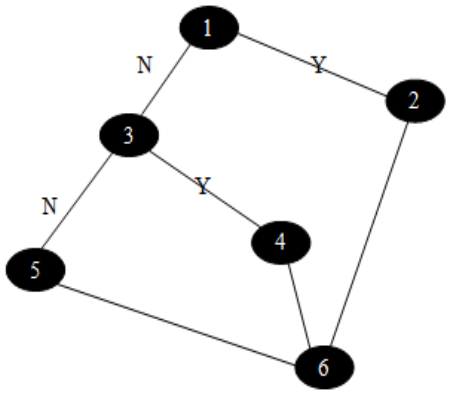

Fig. 2: CFG of the quadratic equation code

SIMULATION RESULTS

The experiment is implemented in MATLAB 2010. The test case prioritization technique's basic evaluation is to have maximum number of faults covered and statement covered with minimum number of test cases required. In this approach, the execution time of every test case is also analyzed. In this example, there are test cases forming Test Suite (TS) = {T1, T2, T3, T4, T5, T6, T7, T8, T9, T10} and the faults covered by those test cases are represented as Faults Covered (FC) = {F1, F2, F3, F4, F5}. Similarly the statements covered by the test cases are denoted as Statements Covered (SC) = {S1, S2, S3, S4, S5, S6, S7}. The Control Flow Graph (CFG) is seen in Fig. 2.

This section compares the performance of the proposed PSABC approach with the other optimization

Table 3: Test case with number of faults covered and execution time taken

Test case/faults	F1	F2	F3	F4	F5	No. of faults covered	Execution time
T1	x		x	x	x	4	9
T2		x	x		x	3	8
T3	x		x		x	3	12
T4			x		x	2	15
T5	x	x		x		3	11
T6	x			x		1	9
T7	x	x		x		3	8
T8	x			x	x	3	7
T9		x		x	x	3	5
T10	x	x	x		x	4	7

Table 4: Test cases with number of statements covered and execution time taken

Test case/faults	S1	S2	S3	S4	S5	S6	S7	No. of faults covered	Execution time
T1	x	x		x		x		4	9
T2	x		x		x			3	6
T3	x	x		x		x	x	5	10
T4		x	x			x	x	4	7
T5		x	x	x	x	x	x	6	11
T6	x					x		2	6
T7	x	x			x	x		4	8
T8	x			x			x	3	4
T9		x	x			x		3	4
T10			x		x	x	x	4	3

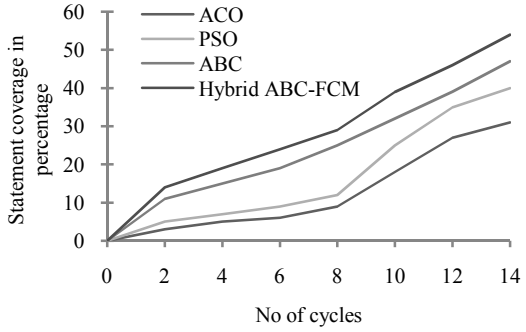

Fig. 3: No. of cycles vs. statement coverage (%) comparison

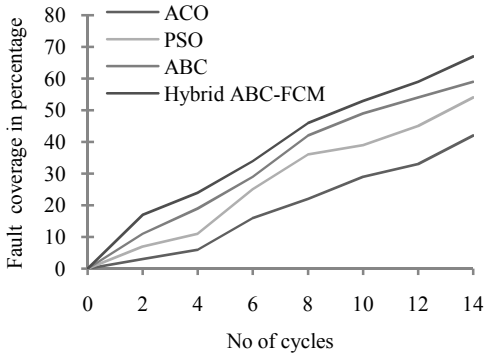

Fig. 4: No. of cycles vs. fault coverage (%) comparison

approaches such as ACO, PSO and ABC, in terms of percentage of statement coverage and fault coverage.

Table 3 and 4 clearly shows the Test cases with the faults and statements covered in particular execution time.

Figure 3 shows the comparison of the number of cycles and the statement coverage in percentage. From the Fig. 3, it is clear that the proposed test case prioritizations approach using hybrid ABC-FCM provides better statement coverage when compared with ACO, ABC and PSO optimization approaches.

Figure 4 shows the fault coverage comparison in percentage for the approaches such as ACO, PSO, ABC and hybrid ABC-FCM. It can be observed from the graph that, there is significant increase in the faults coverage with the increase in the number of cycles. The proposed approach outperforms the other two approaches in terms of the fault coverage.

Thus, it can be observed from the simulation results that the test cases are prioritized based on higher statement coverage and fault coverage using the hybrid ABC-FCM approach.

CONCLUSION

The main focus here is on test case prioritization using swarm intelligence techniques for optimizing the test suite. By prioritizing test cases using swarm intelligence techniques, the cost and time of regression testing could be minimized to a considerable degree. Based on the performance of Swarm intelligence algorithms like ABC and FCM a hybrid algorithm is designed here. Integrating this hybrid with PSO a new algorithm is generated that has potentially the capacity to prioritize test cases and minimize the number of test cases in the test suite. As optimization is the result of reducing test cases, the proposed approach ensures that the existing test suite consists of only those test cases that can cover all statements and paths in the application with a minimal number of test cases. The proposed approach evaluates the optimal test cases based on statement coverage as well as fault coverage with a constraint on execution time. It is observed from the experimental results that the proposed hybrid ABC-FCM based test case prioritization based approach provides better results when compared with ACO, PSO and ABC individually.

REFERENCES

Al-Tabtabai, H. and P.A. Alex, 1999. Using genetic algorithms to solve optimization problems in construction. Eng. Constr. Archit. Manage., 6(2): 121-32.

Bansal, P., 2013. A critical review on test case prioritization and optimization using soft computing techniques. Proceeding of 2nd International Conference on Role of Technology in Nation Building (ICRTNB, 2013), ISBN: 97881925922-1-3.

Binder, R.V., 2000. Testing Object-oriented Systems: Models, Patterns and Tools. Addison-Wesley. Reading, Mass.

Clarke, L.A., 1976. A system to generate test data and symbolically execute programs. IEEE T. Software Eng., SE-2(3): 215-222.

Dervis, K. and C. Ozturk, 2011. A novel clustering approach: Artificial Bee Colony (ABC) algorithm. Appl. Soft Comput., 11: 652-657.

Doungsaard, C., K. Dahal, A. Hossain and T. Suwannasart, 2008. An automatic test data generation from UML state diagram using genetic algorithm. Supported by the EU Asia-link project - TH/Asia Link/004 (91712) -Euro-Asia Collaboration and Networking in Information Engineering System Technology 2008 (EAST-WEST).

Fabrício, G.D.F., L.B.M. Camila, L.D.C. Gustavo Augusto and T.D.S. Jerffeson, 2010. Optimization in software testing using metaheuristics. Revista de Sistemas de Informação da FSMA n., 5(2010): 3-13.

Kaur, A. and S. Goyal, 2011. A bee colony optimization algorithm for fault coverage based regression test suite prioritization. Int. J. Adv. Sci. Technol., Vol. 29.

Korel, B., 1990. Automated software test data generation. IEEE T. Software Eng., 10(8): 870-879.

Krishnamoorthi, R. and S.A. Sahaaya Arul Mary, 2009. Regression test suite prioritization using genetic algorithms. Int. J. Hybrid Inform. Technol., 2(3).

Krishnamoorthi, M. and A.M. Natarajan, 2013. Artificial bee colony algorithm integrated with fuzzy C-mean operator for data clustering. J. Comput. Sci., 9(4): 404-412.

Li, H. and C.P. Lam, 2011. An Ant Colony Optimization Approach to Test Sequence Generation for State based Software Testing. ECU Publications Pre, 2011.

Mahapatra, R.P. and J. Singh, 2008. Improving the effectiveness of software testing through test case reduction. Proceedings of World Academy of Science, Engineering and Technology, 2, ISSN: 1307-6884.

Pressman, R.S., 2007. Software engineering: A practitioners Approach. 6th Edn., Ch. 1(33-47), 13(387-406), 14(420-444). McGraw-Hill, New York.

Reid, S.C., 1997. An empirical analysis of equivalence partitioning, boundary value analysis and random testing. Proceedings of the 4th International on Software Metrics Symposium. Albuquerque, NM, USA, pp: 64-73.

Rothermel, G., M.J. Harrold, J. Ostrin and C. Hong, 1998. An empirical study of the effects of minimization on the fault detection capabilities of test suites. Proceedings of the International Conference on Software Maintenance, pp: 34-43.

Shi, Y. and R. Eberhart, 1998. A modified particle swarm optimizer. Proceedings of the IEEE International Conference on Evolutionary Computation, pp: 69-73.

Sonmez, Y., 2011. Multi-objective Environmental/ economic dispatch solution with penalty factor using artificial bee colony algorithm. Sci. Res. Essays, 6(13): 2824-2831.

Srikanth, A., N.J. Kulkarni, K.V. Naveen, P. Singh and P.R. Srivastava, 2011. Test Case Optimization using Artificial Bee Colony Algorithm. In: Abraham, A. (Ed.), ACC 2011, Part III, CCIS 192. Springer-Verlag, Berlin, Heidelberg, pp: 570-579.

Zhong, H., L. Zhang and H. Mei, 2003. An experimental study of four typical test suite reduction techniques. Inform. Software Tech., 50(6): 534-546.