TEST Engineering & Management

May – June 2020
ISSN: 0193-4120 Page No. 10019- 10032

Selection and Optimization of Automated Test data by using State Chart Diagram and Hybrid Firefly Algorithm

Rajesh Kumar Sahoo[1], Taresh Singh[2], Priyabrata Sahu[3], Bibhuprasad Sahu[4]

[1]Department of Computer Science & Engineering, Ajay Binay Institute of Technology, Cuttack, Odisha, India

[2]Department of Computer Science & Engineering, Pranveer Singh Institute of Technology, Kanpur, Uttar Pradesh, India

[3]Department of Computer Science Engineering & Application, Indira Gandhi Institute of Technology, Sarang, Biju Pattnaik University of Technology, Odisha

[4]Department of Computer Science & Engineering, Gandhi Institute For Technology, Bhubaneswar, Odisha, India

Article Info
Volume 83
Page Number: 10019- 10032
Publication Issue:
May - June 2020

Article History
Article Received*: 19 November 2019*
Revised*: 27 January 2020*
Accepted*: 24 February 2020*
Publication*: 18 May 2020*

Abstract:
Testing of software is used to generate error or bugs. Generation of test cases is a key factor in software testing. Manual testing is a time-consuming and costly process which may generate various errors during software development process. Automated testing reduces the cost and time for generating the test cases. Test case generation is to be adequate requirements of the problem. The proposed approach is used to optimize the test cases obtained from UML state chart diagram using hybrid Bee Colony Firefly Algorithm (BCFA) which is the combination of a bee colony and firefly algorithm. In order to generate the test cases, state chart is diagram was converted into its corresponding intermediate graph form called State Chart Diagram Graph (SCDG). This hybrid technique is used to generate the automated optimized test cases through withdrawal operation of an ATM without dependency. The proposed approach also identifies the operational faults, execution faults and message faults in the present study.

Keywords*: Automated testing, Bee colony algorithm, Firefly algorithm, BCFA approach, generation and optimization of test cases.*

I. INTRODUCTION

Testing plays an important role for developing the software. By using the selected test cases in software testing gives the desired result with less effort. Software testing is one phase of software development life cycle which detects faults or errors for designing the quality software. Testing is done throughout the process of software development [1].Generation of test cases with test data having various merits over test case design through code based testing. Software testing depends on the models because the test cases remain same even some changes occurs in the code. Designing the models is used on the basis of generation of test cases or test data and also it reduces the cost [2].

The optimization techniques are used to design the suitable test case which plays a very crucial role in software development process. It requires the key attributes like correctness and quality for generating the test cases or test data. Automated testing is applied to increase the reliability and test case coverage of the software product. Automated test case design gives the significant reduction in time and effort by increasing the reliability of software by increasing the coverage [6].

D.D.Karaboga [3] introduced Bee colony algorithm in 2005 and by this technique, the honey

May – June 2020
ISSN: 0193-4120 Page No. 10019- 10032

bees are searching different food source position in order to replace the solution with a new improved solution. Dr. Xin She yang developed the firefly algorithm in the year 2007[14]. There are two important variables are used in firefly algorithm which is attractiveness and the intensity of light. One firefly attracts to other fireflies through brighter firefly than itself. So the attractiveness depends on the light intensity. The intensity of light and attractiveness decrease if the distance between fireflies increases [8].

The proposed BCFA approach uses the hybrid technique which is a combination of two techniques like bee colony and firefly algorithm where the test cases are optimized and it inspires software developers to enhance the design quality of software. This paper represents the model based technique which is used for automated generation and optimization of test cases by using bee colony firefly algorithm (BCFA). Through this approach, the automatic test cases are generated using state chart diagram graph (SCDG). This proposed work emphasizes on the appropriate BCFA hybrid optimization technique which gives a better result which is optimal.

The rest of the paper is organized as follows. Section II discusses basics of automated testing, overview of bee colony algorithm, firefly algorithm, and BCFA hybrid algorithm. Section III is for literature survey, Section IV represents the proposed systems, and methodology and working principle of proposed approach. Section V focuses on the simulation results; Section VI focuses on the discussion and future scope and Section VII concludes the paper.

II. BASIC CONCEPTS

1. Overview of automated testing

The automatic test cases or test data takes input from program code and generates the optimal cases by applying different meta-heuristic algorithms [21]. Unified Modeling Language (UML) described as a standard for modeling the behavior of a system. It is used to specifying and modifying the system under development. Modeling of data is emphasized on requirements of data needed. The object model is used to explain the software system through objects. Now UML is used to design and analyzed the large complex systems. Existing approaches are available for automatic generation of test cases but the purposed approach emphasizes to generate the automated best test cases or test data in less time. This proposed paper focuses on the redundancy, selection and optimization challenges of test cases.

2. Overview of Bee Colony Algorithm

Bee colony is the popular algorithm of swarm intelligence technique. It is a nature-based stochastic method which emphasizing on searching for food behavior of honey bees. Possible set of solutions represent the food source position. The amount of nectar symbolizes the fitness values of all solutions or food source. Generally employed, onlooker and scout bees are available in bee colony algorithm. Employed bee initiated the generation of food sources and their fitness functional values are calculated randomly. Through onlooker bee selected food sources are improved to produce better results. Finally the best food source or candidate solution is memorized.

3. Overview of Firefly Algorithm

The Firefly Algorithm is a bio-inspired heuristic algorithm which is a population based stochastic method which is derived and motivated by flashing or mating behavior of fireflies. The light intensity is the key factor where the firefly moves towards the other fireflies. The light intensity less attracts if the distance increases between fireflies and the source of light. The attractiveness of fireflies is having a mutual relation with the brightness. According to the algorithm, the current best solution is represented through the fireflies with high intensity of light or attractiveness. The firefly will move randomly to search new better firefly for the next iteration. The position of all fireflies represents a possible set of solutions and their light intensities represent corresponding fitness functional values.

TEST Engineering & Management

May – June 2020
ISSN: 0193-4120 Page No. 10019- 10032

4. Overview of BCFA

The proposed BCFA hybrid Algorithm is created or developed by merging the Bee Colony Optimization Algorithm with the approach used in firefly Algorithm. Here total population of the candidate solution is subdivided into two parts. One part of the solution undergoes BCO and another part undergoes firefly optimization algorithm. According to the intelligent behaviors, the proposed technique generates the optimal number of test cases which is robust and focusing on generating the possible path from control flow graph. The advantages of this algorithm are for its implementations in complex functions with mixed, random and discrete values.

III. LITERATURE SURVEY

Swain et al. [10] focused on the strategy which gives information after combining the use case and sequence diagram that is used for generating the test cases. Khandai et al. [11] described a technique to generate the test cases from UML combinational diagrams like sequence and activity diagram. An activity diagram is converted into activity diagram graph by applying the criteria of path coverage. Similarly, sequence graph is generated by message path coverage. This paper also explains how the generated test cases are traversed from activity sequence graph (ASG). Monalisha Sharma et al. [9] described an approach to generate the test cases from UML diagrams which is useful in software design. An approach is initiated by Kansomkeat et al. [18] to generate the test sequences by using UML state chart diagrams. The criteria of testing are guiding the test sequence generation which finds the coverage of transition and states using testing flow graph (TFG). Korel [15] introduced a method where test cases or test data are generated by functional testing with minimization and data flow concepts. Test cases or test data are used through actual values of input variables. During program execution the search algorithm locating the selected control path which is traversed. Sahoo et al. [13] explained how the automated test cases are generated and optimized by using different meta- heuristic algorithms like harmony search, particle swarm optimization and bee colony algorithms. According to this paper, bee colony algorithm generates the optimized test cases in very less iteration as compared to harmony search and particle swarm algorithm. Suresh et al. [17] represented that genetic algorithm (GA) is used to generate the test data automatically through basis path testing. According to this paper indicates that GA is more effective and efficient to generate the test data automatically. Sumalatha [16] focused on how the test cases are optimized by UML diagram like activity diagram using evolutionary algorithms. Geniana Ioana Litiu et al. [20] explained how evolutionary algorithms are generating the test path and comparing their results among particle swarm optimization, Genetic algorithm, and simulated annealing. Biswal [4] focused on the test scenarios which are analyzed by sequence diagram. The function is executed by objects in the sequence diagram with the exchange of messages. Kaur et al. [5] focused on how Bee colony optimization (BCO) algorithm generates the test suite in regression testing. In this paper, BCO algorithm is designed for maximum fault coverage with the generation of test cases in less time. Samuel et al. [19] presented a technique to test the object- oriented software which is based on UML sequence diagram and also explained how sequence diagrams generate the test cases by using dynamic slicing technique.

IV. PROPOSED SYSTEM

This paper proposes a methodology for generation of test cases for withdrawal system of an ATM machine from the State Chart Diagram Graph (SCDG). In this approach, the combinational form of a bee colony and firefly algorithm (BCFA) is used to optimize the test cases. In this proposed approach combines the bee colony algorithm emphasizing on the food source position and the initialized with the idea that honey bees will search for better position of food source in the hope to get a better result. Firefly Algorithm (FA) is conceptualized by using two variables like attractiveness and intensities of light through the movements of fireflies to get the best result.

May – June 2020
ISSN: 0193-4120 Page No. 10019- 10032

BCFA is a combination of Bee Colony and Firefly algorithms which may generate the optimum solution. This paper also aims at finding out the effectiveness through various test cases of test data. This method generates and optimizes the test cases with maximizing the path coverage. This method is used for evaluating its efficiency and effectiveness for generating the test cases for maximizing to achieve the goal.

1. **Necessity of Proposed System**

The proposed system is intended to generate an automatic and optimized test case from a UML model with existing approaches of bee colony firefly algorithm (BCFA). Optimized test cases may not be helpful in the testing process. It may be required to differentiate between the various test cases.

2. **Proposed approach and working on proposed approach**

The proposed hybrid Algorithm is created or developed by merging the Bee Colony Optimization Algorithm with the approach used in firefly Algorithm. Here total population of the candidate solution is subdivided into two parts. One part of the solution undergoes BCA and another part undergoes Firefly optimization algorithm. The advantages of this algorithm are for its implementations in complex functions with mixed, random and discrete values.

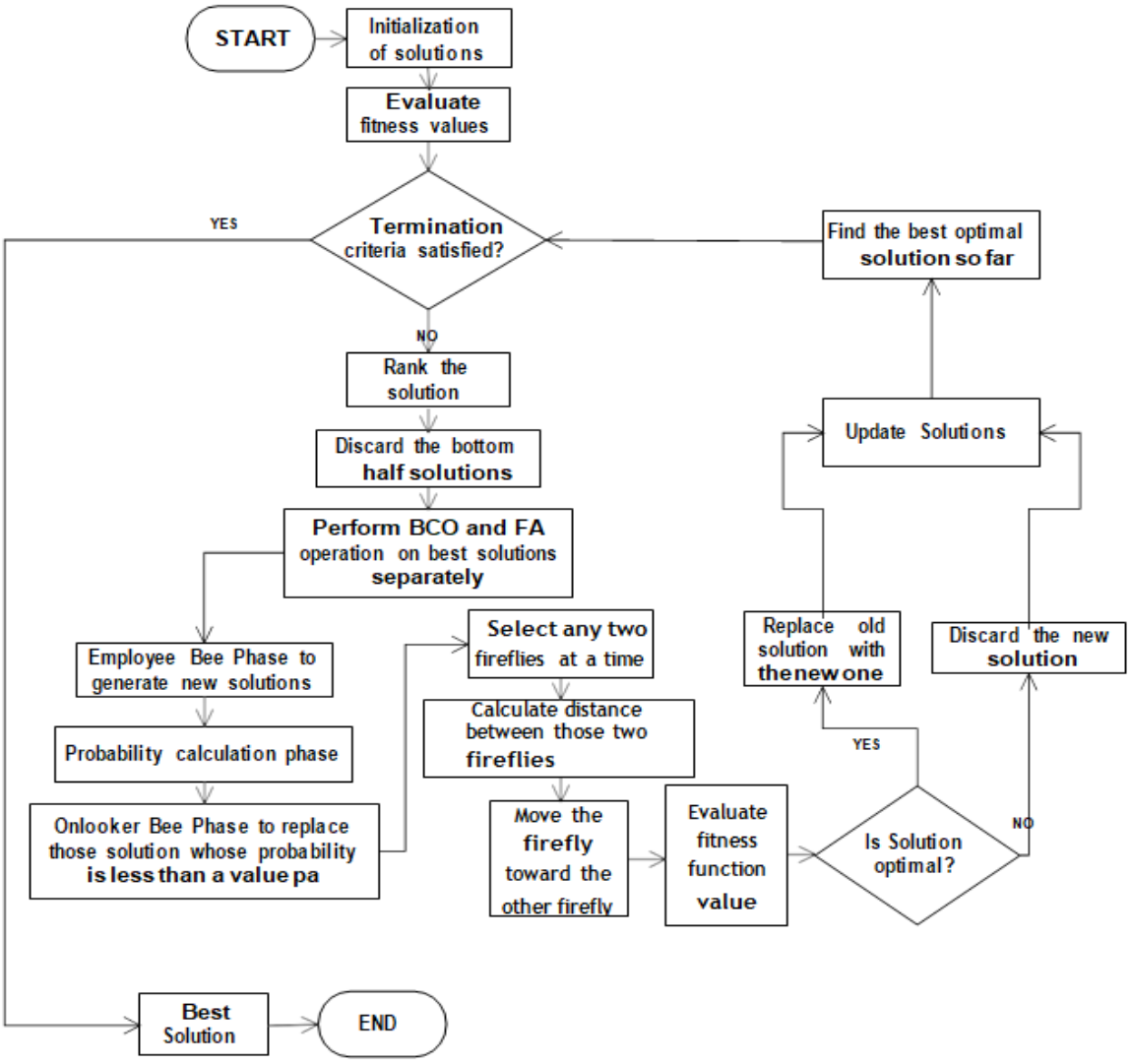

Fig 1: Flowchart of test case generation using BC-FA hybrid approach

May – June 2020
ISSN: 0193-4120 Page No. 10019- 10032

3. **Conversion of State-chart Diagram to State-chart Diagram Graph**

State chart diagram is under UML that describe the time taken by a software system. It consists of mostly transitions and of states. A state represents a condition which satisfies some condition. Start State represents the beginning of a process whereas End State represents the termination of a process. A state transition can be defined as the relationship between two states that indicates transfer of control from one state to another depending upon certain conditions. It also represents the different states and the events that effect to changing the state. "Fig. 2" represents the state chart diagram and state chart diagram graph of withdrawal task of an ATM. Table 1 represents the dependency table for overall operation of ATM which is shown in state chart diagram graph.

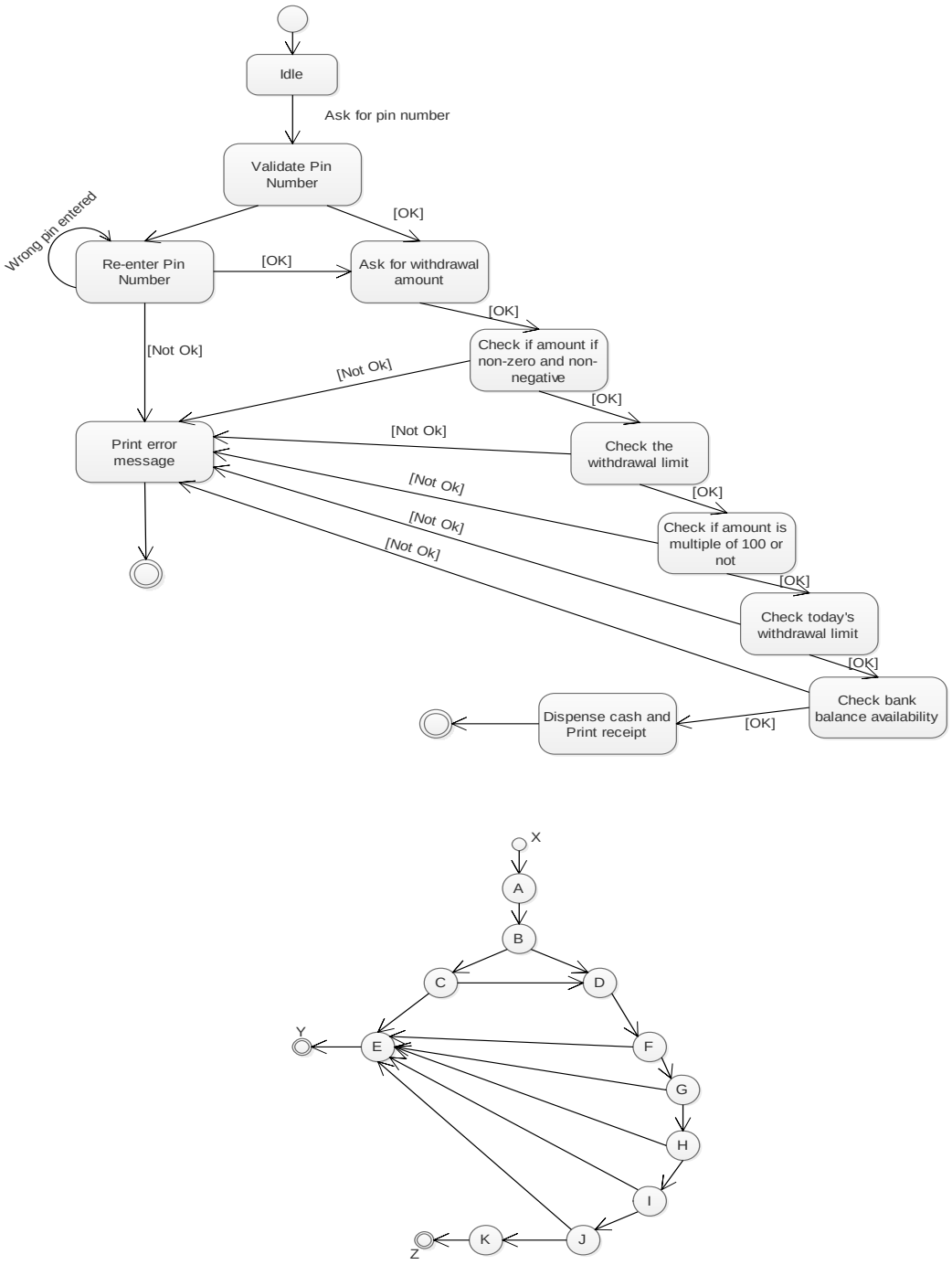

Fig 2. State Chart Diagram and State Chart Diagram Graph of withdrawal operation of an ATM

TEST Engineering & Management

May – June 2020
ISSN: 0193-4120 Page No. 10019- 10032

TABLE I: Dependency Table of State chart Diagram Graph Used in ATM Operation

Node	Activity Name	Possible number of outputs	Dependency Nodes	Input	Expected outputs
A	Ask for PIN	1(B)	NA	User prompts to enter PIN	B: PIN is forwarded for validation
B	Check validity of PIN	2 (C,D)	A	PIN provided by the user	D: Valid PIN
					C: Invalid PIN
C	Re-enter PIN	2 (D,E)	B	Incorrect PIN message	E: Message displayed for incorrect PIN
					D: Valid PIN
D	Ask for withdrawal amount	1 (F)	B,C	User prompts to enter withdrawal amount	F: Amount is forwarded to be checked
E	Print error message	1 (Y)	C,F,G,H,I,J	Invalid input	Y: Message displayed for incorrect withdrawal amount
F	Check if amount is non-negative or non-zero	2 (E,G)	D	Amount entered by the user	G: Amount is forwarded for further checking
					E: Invalid amount
G	Check the withdrawal limit	2 (E,H)	F	Amount entered by the user	H: Amount is forwarded for further checking
					E: Invalid amount
H	Check if amount is a multiple of 100 or not	2 (E,I)	G	Amount entered by the user	I: Amount is forwarded for further checking
					E: Invalid amount
I	Check today's withdrawal limit	2 (E,J)	H	Amount entered by the user	J: Amount is forwarded for further checking
					E: Invalid amount
J	Check bank balance availability	2 (E,K)	I	Amount entered by the user	K: Amount is checked
					E: Invalid amount
K	Dispense cash and print receipt	1 (Z)	J	Amount entered by the user	Z: Cash is dispensed and receipt is printed

X is defined as the starting node. Y and Z are defined as the end nodes where Y is the end node with Unsuccessful result and Z is the end node with the Successful result.

A, B, C, D, E, F, G, H, I, J and K are the intermediate nodes describing various operations or activities occurring within the system during operation execution. Out of which A, B, C, D, E and K describes the whole ATM operation. Rest nodes F to J describe only the withdrawal amount checking operation.

TEST Engineering & Management

May – June 2020
ISSN: 0193-4120 Page No. 10019- 10032

4. Generation and Optimization of test data

After creating SCDG graph, next step is to generate and optimize the test cases. For optimization purpose various meta-heuristic evolutionary algorithms are used. In this proposed approach the hybrid bee colony optimization algorithm is used for optimizing the test cases. The test coverage criteria are calculated through test cases which covered a number of elements. The generations of test cases are reduced.

5. BCFA (Pseudo code for test case or Test data generation by using BCO-FA Hybrid Approach)

Identify all the paths P= {path1, path 2, path3, ….., path n} from starting node to end node
Assign each node in the graph based upon the importance or priority of each node in that graph.
Now apply BCFA to the State Chart Diagram Graph
Calculate the fitness value of each path of the given graph.

$fx=1/(abs(net\_bal-wd\_amt)*min\_bal)+\varepsilon)^2$

where, ε varies from 0.1 to 0.9

Choose the initial best solution, sort the population based upon the fitness function value.
While generation(t)<500
do Rank the solutions
Discard the bottom half solutions having worst fitness values
Top half best solutions undergo operation in two phases separately
Make two copies of best solutions.
One copy undergoes Bee Colony Optimization i.e., Phase1
Another copy undergoes Firefly Algorithm i.e., Phase 2
Phase 1
//Employed Bee Phase
Produce new candidate solution Check the boundary conditions
Evaluate its fitness value
If(finess(new)>fitness(old))
then replace the older solution
//Probability Calculation Phase
Calculate the probability of occurrence of each solution P
//Onlooker Bee Phase If P>rand ()
Produce new candidate solution
Check the boundary conditions
Evaluate its fitness value
If(finess(new)>fitness(old))
then replace the older solution
End If
End If

Phase 2

Select any two fireflies at a time

Calculate distance between those two fireflies

Move the firefly toward the other firefly

Intensity Values / Fitness Function value calculation

Check the boundary conditions

Evaluate its fitness value
If(fitness(new)>fitness(old))
then replace the older solution
End If

Memorize the best solution Update Post and Best values
Generation (k) = Generation(k) +1
Get the best solution so far
End While

Select the best solution with best fitness value from the above pseudo code of BCFA approach

Phase 1 :- (Bee Colony Algorithm)
The new solution can be calculated as

$$c=x(j)+ebf*x(j) \quad (1)$$

where, x(j)= candidate solution at j^{th} position
ebf = random value within the range of [-1,+1]

TEST Engineering & Management

May – June 2020
ISSN: 0193-4120 Page No. 10019- 10032

The probability of occurrence for each candidate solution is calculated as follows:

prob(j)=fx(j)/tfx; (2)

Where, prob = probability factor
fx(j) = fitness function value
tfx = total fitness value of all candidate solution

In Onlooker Bee phase, the solution having probability greater than a random value within the range of [0, 1] are selected and their corresponding solutions are improved with the help of the following equation:

v(j)=x(j)+ebf*x(jj); (3)

where, ebf = random value in the range of [- 0.1,+0.1]

Phase 2 :- (Firefly Algorithm)
The distance between any two fireflies can be calculated as

r =(x (i)-x(j))^2/vmin (4)

where, x(j)= candidate solution at j^{th} position
x(i)= candidate solution at i^{th} position
vmin= minimum balance

The movement of a firefly toward another is calculated as follows:

x1=x(i)+round((beta0*(exp(- (gamma*r)))*(x(j)-x(i))+vmin*alpha*(rand-0.5)) (5)

where, x1=new solution
rand= random value within the range of [0,1]
beta0 = attractiveness at r=0
gamma and alpha = firefly parameters where beta0=1, gamma=1 and alpha=0.2

6. Methodology

For Mathematical function,

$f(x)=1/(abs(suc\_bal)+\varepsilon)^2$ (6)

where, $0.1 <= \varepsilon < 0.9$ (taking ε -value because overflow condition due to infinity).

Here Successive Amount (suc_amt) is defined as:

suc_bal = net_bal - (wtd_amt-min_bal) (7)

where, net_bal = current account balance
min_bal= minimum bank balance limit

Initially, the population size and the number of generations or a maximum number of iterations is provided by the user. After that, an initial population is generated randomly and their corresponding fitness values are calculated and stored. The initial best optimal solution is calculated. Then the candidate solutions are sorted in terms of their fitness values. Higher the fitness value more the solution tends toward optimality. After the sorting operation, the bottom half worst solutions is discarded and are replaced with a copy of top half best solution found so far. Then both copies of top half best solution undergo two different phases of optimization techniques. In this case the first phase i.e., in Phase 1, the candidate solutions undergoes Bee Colony Optimization (BCO) and another copy of candidate solution undergo the second phase (Phase 2) i.e., Firefly Algorithm (FA) optimization. Phase 1 of BCO is subdivided into two more phases i.e., Employed Bee phase and Onlooker Bee phase. In Employed Bee phase a new solution is generated and checked if the fitness function values of the new candidate solution are better than the old existing solution or not. If the solution is found to be producing better solution than the old solution replaced is replaced by the new solution. After Employed Bee Phase, the relative fitness value of each candidate solution

May – June 2020
ISSN: 0193-4120 Page No. 10019- 10032

is calculated. In Onlooker Bee phase, the candidate solutions having a relative value less than a specific constant value 'pa' then that solution is discarded from the memory and is replaced with a newly generated random solution. In Phase 2, two candidate solutions or fireflies are selected at random and distance between those two fireflies is calculated. Then the firefly having lower fitness value is moved toward the firefly having higher fitness value. If the fitness function value of the new candidate solution is better than the old existing solution then the old solution is replaced with the new solution. Then new better solutions are created. After the completion of two phases of optimization, the current best solution is memorized. The result gained from both phases is merged. Again all the candidate solutions are sorted and the bottom half worst solution is discarded and are replaced with a copy of top half best solution. Then both copies of top half best solution undergo two phases and the programs iterates until termination criteria is satisfied. The solution produced so far is the optimal solution.

According to Firefly Algorithm, the position of all fireflies represents a possible set of solutions and their distance from the light intensity which represents fitness values or quality of all solutions. In BCFA hybrid approach combined the Bee Colony and Firefly algorithm which gives the optimal solution to maximize the mathematical function f(x).It may be implemented using MATLAB-7.0 as shown in Table-1. This table primarily focuses on generating the best solution in the search space.

TABLE II: Fitness Function Value for Each Sample Space or Test Case

Iteration Number	Bee Colony Algorithm(BCA)		Firefly Algorithm(FA)		Bee Colony Firefly Algorithm (BCFA)	
	Test data	Fitness Function Value	Test data	Fitness Function Value	Test data	Fitness Function Value
1	4100	5.9779e-010	3900	5.9199e-010	4000	6.25e-010
10	6200	6.6425e-010	5600	6.4418e-010	9000	7.716e-010
20	7300	7.0358e-010	8300	7.4245e-010	11500	8.9106e-010
30	9200	7.8024e-010	10800	8.5496e-010	14900	1.1037e-009
40	15600	1.1569e-009	13400	1.0014e-009	18100	1.3819e-009
50	19400	1.5259e-009	15400	1.1413e-009	20900	1.7217e-009
60	20400	1.6524e-009	18000	1.3717e-009	24400	2.3564e-009
70	23400	2.1433e-009	20500	1.6659e-009	27200	3.1561e-009
80	27600	3.3029e-009	22900	2.0474e-009	30300	4.6276e-009
90	30400	4.6912e-009	25100	2.5252e-009	33100	7.0615e-009
100	34300	8.7341e-009	28200	3.543e-009	36400	1.352e-008
110	36900	1.5241e-008	30900	5.0299e-009	39400	3.1886e-008
120	39000	2.7776e-008	33200	7.1817e-009	42900	2.2673e-007
130	41100	6.5741e-008	36100	1.2624e-008	44000	9.964e-007
140	42700	1.8901e-007	38900	2.6873e-008	44000	9.978e-007
150	43500	4.4435e-007	41200	6.9244e-008	44000	9.978e-007
160	44000	9.9968e-007	43500	4.4429e-007	44000	9.989e-007
170	44000	9.9989e-007	44000	9.978e-007	44000	9.998e-007
180	44000	9.9997e-007	44000	9.998e-007	44000	9.998e-007
200	44000	9.998e-007	44000	9.998e-007	44000	9.998e-007

May – June 2020
ISSN: 0193-4120 Page No. 10019- 10032

V. SIMULTION RESULTS

The proposed approach generates the automated test cases or test data for Bank ATM by using BCFA hybrid algorithms. The figure-3 shows the relation between two variable quantities which are iteration numbers and test cases or test data measured along one of a pair of axis represented in table-1.

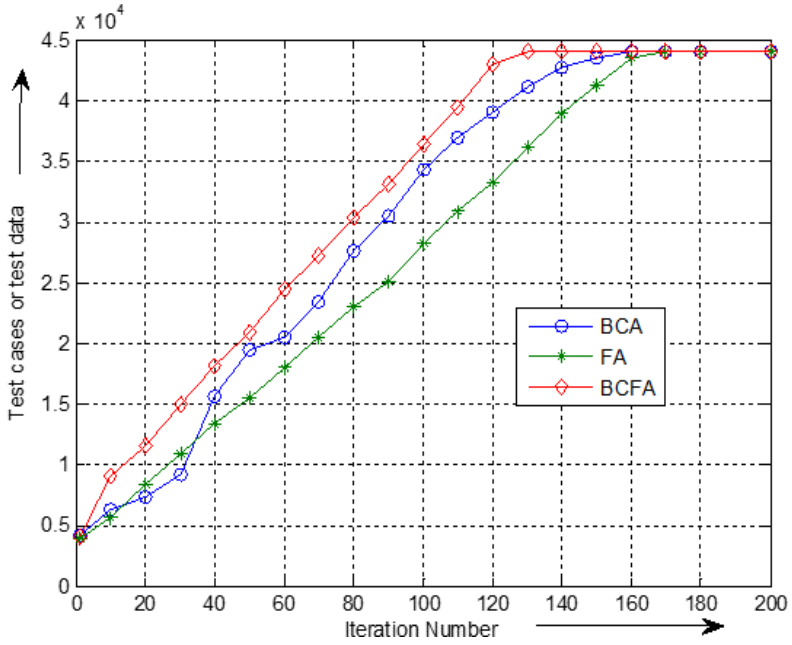

Fig 3. Graphical representation of iteration numbers test data

After evaluation, it was found that using bee colony Optimization technique the optimal solution is achieved after 160 iterations whereas by using firefly Algorithm the optimal solution is achieved after 170 iterations. But by implementing the hybrid approach (BCFA) it was observed that the optimal result is achieved much earlier around 130 iterations. The proposed approach generates the test case or test data for Bank ATM's withdrawal operation using bee colony, firefly and BCFA algorithm. Every generated test case traverses test paths that contain a set of nodes which is the subset of the original set of nodes. The reduced test cases cover all nodes and edges with traversing the path by DFS algorithm which is implemented in MATLAB 7.There are seven possible numbers of paths that can be traversed through BFS by using SCDG system graph. Out of which only one path produces an optimal result and rest of the paths do not produce optimal result.

Here Path number 1, 2 3, 4, 5, 6, 7, 8, 9, 10, 11 and 12 all produces incorrect result and can be labeled as unsuccessful. Only Path 13 produces correct optimized solution and can be regarded as Successful. Table 3 represents the possible paths generated by SCDG.

May – June 2020
ISSN: 0193-4120 Page No. 10019- 10032

Table III: Possible Path Generation by Using State Chart Diagram Graph

<Path 1	<Path 2	<Path 3	<Path 4	<Path 5	<Path 6	<Path 7
State X	State X	State X	State X	State X	State X	State X
A(m1,a,b)	A(m1,a,b)	A(m1,a,b)	A(m1,a,b)	A(m1,a,b)	A(m1,a,b)	A(m1,a,b)
B(m2,b,c)	B(m2,b,c)	B(m2,b,c)	B(m2,b,c)	B(m2,b,c)	B(m2,b,c)	B(m2,b,c)
C(m3,c,b)	D(m4,c,b)	C(m3,c,b)	D(m4,c,b)	C(m3,c,b)	D(m4,c,b)	C(m3,c,b)
E(m5,b,d)	F(m6,b,c)	D(m5,b,c)	F(m6,b,c)	D(m4,c,b)	F(m6,b,c)	D(m4,c,b)
State Y >	E(m5,c,d)	F(m6,b,c)	G(m7,b,c)	F(m6,b,c)	G(m7,b,c)	F(m6,b,c)
	State Y >	E(m5,c,d)	E(m5,c,d)	G(m7,b,c)	H(m8,b,c)	G(m7,b,c)
		State Y >	State Y >	E(m5,c,d)	E(m5,c,d)	H(m8,b,c)
				State Y >	State Y >	E(m5,c,d)
						State Y >

<Path 8	<Path 9	<Path 10	<Path 11	<Path 12	<Path 13
State X	State X	State X	State X	State X	State X
A(m1,a,b)	A(m1,a,b)	A(m1,a,b)	A(m1,a,b)	A(m1,a,b)	A(m1,a,b)
B(m2,b,c)	B(m2,b,c)	B(m2,b,c)	B(m2,b,c)	B(m2,b,c)	B(m2,b,c)
D(m4,c,b)	C(m3,c,b)	D(m4,c,b)	C(m3,c,b)	D(m4,c,b)	C(m3,c,b)
F(m6,b,c)	D(m5,b,c)	F(m6,b,c)	D(m5,b,c)	F(m6,b,c)	D(m5,b,c)
G(m7,b,c)	F(m6,b,c)	G(m7,b,c)	F(m6,b,c)	G(m7,b,c)	F(m6,b,c)
H(m8,b,c)	G(m7,b,c)	H(m8,b,c)	G(m7,b,c)	H(m8,b,c)	G(m7,b,c)
I(m9,b,c)	H(m8,b,c)	I(m9,b,c)	H(m8,b,c)	I(m9,b,c)	H(m8,b,c)
E(m5,c,d)	I(m9,b,c)	J(m10,b,c)	I(m9,b,c)	J(m10,b,c)	I(m9,b,c)
State Y >	E(m5,c,d)	E(m5,c,d)	J(m10,b,c)	K(m11,c,d)	J(m10,b,c)
	State Y >	State Y >	E(m5,c,d)	State Y >	K(m11,c,d)
			State Y >		State Z >

VI. DISCUSSION AND FUTURE SCOPE

By considering the mathematical function $fx=1/(abs(net\_bal- wd\_amt)+\varepsilon)^2$,where ε varies from 0.1 to 0.9,this proposed paper generates and optimized the test cases as well as test data automatically through bee colony algorithm, firefly algorithm and combinations of a bee colony and firefly algorithm(BCFA). By considering some sample test cases it has been observed that the functional value depends on upon the parametric values of the input variables and food source position. Firefly algorithm generates the optimized test cases with test data which would also have the coverage of path. BCFA hybrid approach generates the test cases or test data with less time as compared to other two approaches like bee colony and firefly algorithm. The position of all fireflies represents a possible set of solutions and their light intensities represent corresponding fitness values or quality of all solutions. This proposed BCFA algorithm is optimized the test cases. For any algorithm implementation first, the algorithm is converted into pseudo code before the application developed. Firefly algorithm is determined with maximizing the path covered. The optimum value is obtained from Firefly algorithm comes from Bee Colony Algorithm (BCA). In future different hybrid search based optimization technique like PSBCA, GFA, and BCBA may be used for generating the test cases from combinational UML diagrams with path coverage which also improve the software design quality.

May – June 2020
ISSN: 0193-4120 Page No. 10019- 10032

VII. CONCLUSION

This proposed technique is used for generation and optimization of test cases or test data by removing the ambiguities' effectively and efficiently. The proposed system takes less CPU execution time to choose the best test path which is more efficient and reliable for the development of software. According to experimental results, the proposed BCFA hybrid approach gives a better result, takes less CPU execution time and minimized the error in less iteration as compare to bee colony algorithm (BCA) and Firefly Algorithm (FA).

REFERENCES

[1] A. Bertolino, "Chapter 5: Software testing", IEEE SWEBOK Trial Version 1.0, May 2001.

[2] S. Srivastava, S. Kumar and A. K. Verma, "Optimal path sequencing in basis path testing", International Journal of Advanced Computational Engineering and Networking, ISSN (PRINT):2320- 2106, Vol.1, No.1, 2013.

[3] Basturk, B., Karaboga, D.," A powerful and efficient algorithm for numerical function optimization: artificial bee colony(ABC) algorithm", In Proceedings of the IEEE Swarm Intelligence Symposium, pp. 459–471. IEEE, 2006.

[4] Biswal Baikuntha Narayan , Test case Generation and Optimization of Object-oriented Software using UML behavioral Models,2010, http://ethesis.nitrkl.ac.in/2923/

[5] Dr. Arvinder Kaur and Shivangi goyal, "A Bee Colony optimization Algorithm for fault coverage based regression test suite Prioritization" International Journal of Advanced Science and Technology, Vol.29, April 2011.

[6] P.N. Boghdady, N.L. Badr, M. Hashem and M.F. Tolba, "A proposed test case generation technique based on activity diagrams, "International Journal of Engineering & Technology, IJET- IJENS,Vol.11, No,3.

[7] S.Anand et al., "An Orchestrated Survey on Automated Software Test Case Generation", Journal of Systems and Software, 2013.

[8] Rajesh Kumar Sahoo, Durga Prasad Mohapatra, Manas Ranjan Patra, "A firefly Algorithm Based Approach for Automated Generation and Optimization of Test cases", International Journal of Computer Sciences and Engineering,vol.4,Issue-8,2016.

[9] Monalisha Sharma , Debasish Kundu ,Rajib Mall, "Automatic test case generation from UML sequence diagrams", 15th international conference on advanced computing and communications PP. 60-64.

[10] Santosh Kumar Swain, Durga Prasad Mohapatra and Rajib Mall, "Test case generation based on use case and sequence diagram, International journal of software Engineering , IJSE Vol.3, No-2, 2010.

[11] M. Khandai , A. A. Acharya , D. P. Mohapatra, Test case generation for a concurrent system using UML combinational diagram , International journal of Computer Science and Information Technologies,2011.

[12] Supaporn Kansomkeat , Jeff offutt, Aynur Abdurazik, Andrea Baldini , A comparative Evaluation of tests generated from different UML diagrams ", Technical Report, George Mason university , 2008.

[13] Rajesh Kumar Sahoo, Deeptimanta Ojha, Durga Prasad Mohapatra, Manas Ranjan Patra, "Automated Test case Generation and optimization: A Comparative Review", International Journal of Computer Science & Information Technology, Vol.8, No.5, 2016.

[14] X. S. Yang, Firefly Algorithm: Stochastic Test Functions and Design Optimization, International Journal of Bio- Inspired Computation, Vol. 2, No. 2, pp.78–84, 2010.

May – June 2020
ISSN: 0193-4120 Page No. 10019- 10032

[15] Korel, B.,"Automated software test data generation", IEEE, Trans. Software Engineering, Vol. 16, No.8, pp.870-879, 1990.

[16] V. Sumalatha, "Model-Based Test Case Optimization of UML Activity Diagrams using Evolutionary Algorithms", International Journal of Computer Science and Mobile Applications, Vol.2, No.11, pp.131- 142, 2014.

[17] Yeresime Suresh, Santanu Ku. Rath, "A genetic Algorithm based approach for test data generation in basis path Testing", International Journal of Soft computing and Software Engineering (JSCSE), Vol.3, No.3, 2013.

[18] Kansomkeat. S and Rivepiboon. W,2003, "Automated generating test case using UML state chart diagrams", In proc. SAICSIT 2003, ACM, PP. 296-300, 2003.

[19] Philip Samuel, Rajib Mall and Sandeep Sahoo, "UML sequence diagram based testing using slicing", IEEE conference on Software Engineering, PP. 176-178,2005.

[20] Latiu Geniana Ioana, Cret Octavian Augustin, Vcariu Lucia, "Automatic Test data Generation for software path testing using evolutionary algorithms", International conference on emerging intelligent data and web technologies, Buch a rest , PP.1-8,2012.

[21] R.K. Swain, V.Panthi , and P.K. Behera , "Generation of test cases using activity diagram, "International Journal of computer science and informatics,Vol.3, 2013.