I.J. Intelligent Systems and Applications, **2017, 6, 43-54**
Published Online June 2017 in MECS (http://www.mecs-press.org/)
DOI: 10.5815/ijisa.2017.06.05

Model Driven Test Case Optimization of UML Combinational Diagrams Using Hybrid Bee Colony Algorithm

Rajesh Ku. Sahoo
ABIT, Department of Computer Sc. & Engineering, Cuttack, Odisha, India
E-mail: rajesh_sahoo@rediffmail.com

Santosh Kumar Nanda
Centre for Research, Development and Consultancy, EAST, Bhubaneswar, Odisha, India
E-mail: santoshnanda@live.in

Durga Prasad Mohapatra and Manas Ranjan Patra
NIT, Department of Computer Sc. & Engineering, Rourkela, Odisha, India
E-mail: durga@nitrkl.ac.in
Berhampur University, Department of Computer Sc. & Engineering, Berhampur, Odisha, India
E-mail: mrpatra12@gmail.com

Abstract—To detect faults or errors for designing the quality software, software testing tool is used. Testing manually is an expensive and time taking process. To overcome this problem automated testing is used. Test case generation is a vital concept used in software testing which can be derived from requirements specification. Automation of test cases is a method where it can generate the test cases and test data automatically by using search based optimization technique. Model-driven testing is an approach that represents the behavioral model and also encodes the system behavior with certain conditions. Generally, the model consists of a set of objects that defined through variables and object relationships. This piece of work is used to generate the automated optimized test cases or test data with the possible test paths from combinational system graph. A hybrid bee colony algorithm is proposed in this paper for generating and optimizing the test cases from combinational UML diagrams.

Index Terms—Software testing, automated generation of test cases, model-driven testing, UML diagrams, hybrid bee colony algorithm.

I. Introduction

Software testing is a very important technique to design the faultless software. It takes around 60% of resources for the software development. [18].This technique is also satisfying all the requirement of the customer verification and validation of software is done through (SUT). The main objective is to control the software where it may be automatic and optimized. Automated testing is used to control the test execution with a precondition.Test cases are generated through testing and optimization of software [18]. Test cases may be defined to collect the required data input, performing actions and producing the desired output. Generations of test cases are used to identify test cases with resources and also identify critical domain requirements. Test case development accumulates with requirement specification in a particular path of a program. For finding more faults which are including the software and to minimize the test cases needs to execute and optimize [1].

Model-driven testing is an approach that represents the behavioral model and also encodes the system behavior with certain conditions. It extracts test cases from different models which are built from coding [20].Automated testing is a model based testing where the system model generates test cases automatically. UML is a modeling language which specified for analysis and design of software. It is important for the design of test cases, reduce cost and improve the quality of software [23]. UML specifies the behavior and structure of the system.UML provides the structural analysis of the higher level system.

Kennedy and Eberhart introduced particle Swarm Algorithm which is a search based optimization technique in the year 1995[6].The solutions of this problem are represented through n-dimensional space. The various particles are set and move randomly in a space. In every iteration, particles know their fitness value in the current position and their neighbors' which gives the better fitness functional value. This technique simulates the behavior of birds flocking. The group of particles like birds is searching their food randomly and the particles do not know where the food is available. All particles are having velocities and fitness functional values.

Karaboga introduced bee colony algorithm (BCA) in the year 2005[4, 5]. It is a search based optimization method which simulates the searching for food behavior

of honey bees. It also chooses the best quality food source with the position until the stopping criteria are meeting. Honey bees are looking the search space for generating the feasible solution.BCA possess an ability to find high-quality solutions of difficult combinational problems with the reasonable amount of time.

The rest of the paper is organized as follows. Section II discusses basics of software testing, model-driven testing, an overview of particle swarm optimization algorithm(PSOA), bee colony algorithm(BCA) and hybrid bee colony algorithm(PSBCA).Section III is for related work, and Section IV represents the proposed systems, working principle, generation, and optimization of test cases by using PSBCA and methodology of the proposed approach. Section V focuses on the simulation results; Section VI represents the case study of withdrawal task of an ATM with possible path generations by using SCSEDG system diagram graph. Section VII focuses on the discussion and future scope and Section VIII concludes the paper.

II. Basic Concepts

Model-driven testing (MDT) is done through models of software which used for test data or test case generation. Models give information about the structural aspect of the system.MDT is implemented through different diagrams like sequence, activity, state chart, collaboration diagrams etc. It takes the test data from program code as input and generates the data automatically. Optimization is essential for the applications which used in real-time. This technique is used to generate the best solution from the set of alternatives available. The best solution gives better performance with specific constraints, maximize the resources and minimize the cost.

The particle swarm algorithm (PSA) is a stochastic method that motivated by social behavior of grouped migrating birds. In PSA swarm is the combination of different particles having a candidate solution. Each article usually possesses its current position, current velocity, and its own best position called as pbest. Pbest is the personal best position explored and also incorporates gbest that global best position achieved by all its individuals. The velocity of each particle is dependent on its neighbors which give the best fitness functional value. The fitness value of the particle may be calculated according to function. If the fitness of the particle is better than its previous value, then the position of the particle may be updated to its personal best position. Again if the value of pbest is better than gbest position, the position of global best of the particle is updated.

Bee colony algorithm (BCA) is a stochastic method. The behavior of bees is manipulated through this algorithm. A possible set of solutions represents 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. According to employed bee new position of the food source is generated randomly. Through onlooker bee selected food sources are improved to produce better results. Finally, the best food source or candidate solution is memorized.

The proposed hybrid PSBCA Algorithm is created or developed by merging the particle swarm Optimization Algorithm with the approach used in bee colony Algorithm. Here total population of the candidate solution is subdivided into two parts. One part of the solution undergoes Particle swarm and another part undergoes bee colony algorithm. The advantages of this algorithm are for its implementations in complex functions with mixed, random and discrete values.

III. Related Work

According to Jones [10], automatic test cases or test sets are generated using a Genetic algorithm with predefined criteria of testing and this criterion is set by the requirements. It obtains the branch coverage's with a number of iterations. N. Narmada and D.P.Mohapatra[11] Described some review process which is already developed and some application areas which is developed in near future related to particle swarm optimization. Arvinder Kaur[13] focused on how the bee colony optimization(BCO) algorithm generate 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. Khandai et al. [16] described a technique that generates the test cases from UML combinational diagrams like sequence diagram and activity diagram. An activity diagram is converted into activity graph by applying the criteria of path coverage. Similarly, sequence graph is generated by message path coverage. This paper also explains how the test cases are generated and traversed from activity sequence graph (ASG). Sahoo et al. [22] explained how the automated test data are generated by harmony search algorithm by taking an example of ATM withdrawal operation. Swain et al.[17] focused on the strategy which gives information after combining use case and sequence diagram for generating the test cases. Samuel et al.[21]presented a methodology to test the object-oriented software which is based on UML sequence diagram.This paper also explained how sequence diagrams generate the test cases by using dynamic slicing technique. Shanthi et al. [2] focused on the generation and optimization of test cases through sequence diagram by applying a genetic algorithm which is validated. Generating the test cases from sequence diagram extracts the information from a sequence dependency Table (SDT) and possible test path is generated. Sahoo et al.[9] explained the various nature-inspired meta-heuristic algorithms and their performance. Supapornkansomkeat et al. [15] presented the automatic testing technique which is used to generate test cases from TFG (Testing Flow Graph) without error. Generated test cases are traversed by flow graph criteria and the state transition diagram. Lastly, mutation of test cases is determined. Sahoo et al.[14] focused on how the test

cases are generated and optimized by different meta-heuristic algorithms like harmony search, Particle swarm optimization, and bee colony algorithms. In this paper, bee colony algorithm generated the optimized test cases efficiently with less iteration in comparison to HSA and PSOA. L.T.Bui,et al.[12] explained the aspects of developments, applications, and resources which are related to particle swarm optimization. According to Abdurazik [3], testing criteria is based on the collaboration diagrams for static checking and dynamic testing. It does not generate the test cases which achieve the transition path coverage through system testing graph. Sahoo et al. [19] described how the automated test cases are generated and optimized by firefly algorithm through the withdrawal operation of an ATM. Korel [8] introduced a method where test cases or test data are generated by functional testing with minimization and data flow concepts with valid input variables. During program execution, the search algorithm locating the selected control path which is traversed. An approach is initiated by Kansomkeat[7] for generating test sequences using 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). Sahoo et al. [24] focused on the automated course timetable generation and optimization by a hybrid firefly approach which takes less iteration to optimize in comparison to bee colony algorithm and firefly algorithm.

IV. Proposed Approach

This paper proposes a methodology for generation of test cases for withdrawal system of an ATM machine from combinational system diagram graph which combines the state chart and sequence diagram graph (SCSEDG).Test cases are optimized by particle swarm bee colony algorithm (PSBCA) and generate maximize of path coverage. This method is used for evaluating its efficiency and effectiveness for generating the test cases and for maximizing to achieve the goal.

Necessity of Proposed System

The proposed system is intended to generate an automatic and optimized test case from a hybrid bee colony algorithm through the combinational model. Optimized test cases may not be helpful for the testing process. It may be required to differentiate among the various test cases. First of all the system may be initialized with food source positions. Each food source maintains its own prevailing location on the basis of which the test cases may be generated. This paper also finds out the effectiveness of the proposed approach in the case of a number of generations or test cases.

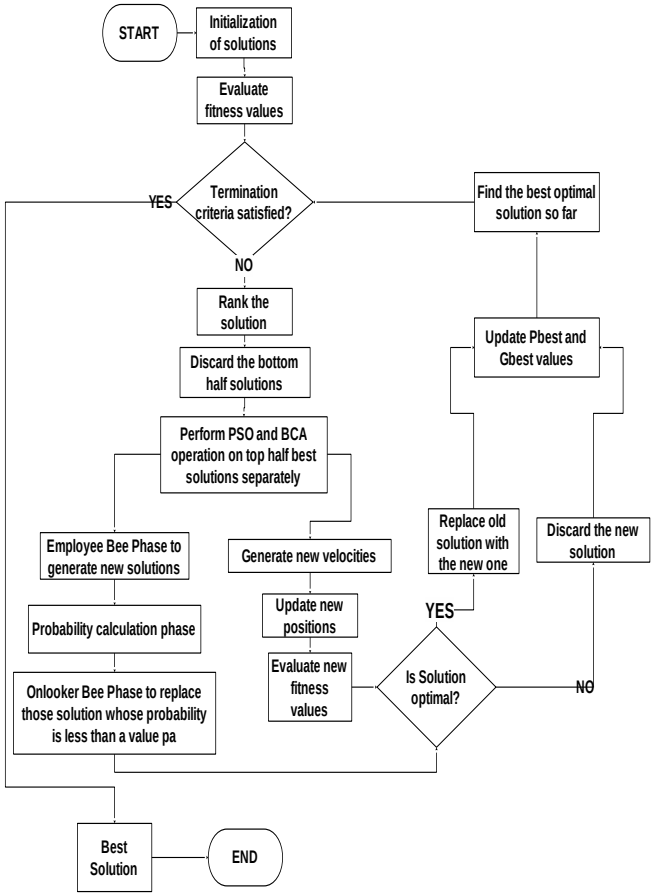

Fig.1. Flowchart of test case generation using PS-BCA hybrid approach

Proposed approach and working on proposed approach

The proposed hybrid Algorithm is created or developed by merging the Particle swarm Optimization Algorithm with the approach used in bee colony Algorithm. Here total population of the candidate solution is subdivided into two parts. One part of the solution undergoes particle swarm algorithm and another part undergoes bee colony algorithm. The advantages of this algorithm are for their implementation in complex functions with mixed, random and discrete values. The flow chart of hybrid Particle Swarm Optimization and Bee Colony Optimization is depicted in "Fig.1".

Conversion of Statechart Diagram to Statechart 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. 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.

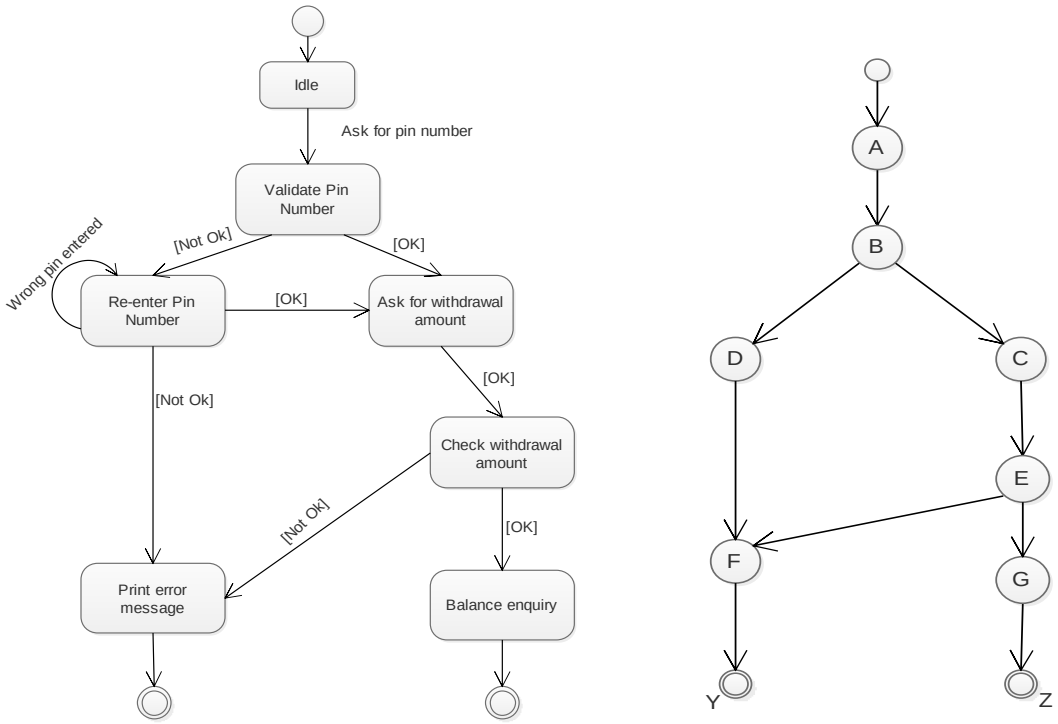

Fig.2. Statechart diagram and state-chart diagram graph of overall operation of an ATM

Table 1. Dependency table of overall operation of an ATM through state-chart diagram graph

Symbol	Activity Name	Possible number of outputs	Dependency	Input	Expected outputs
A	Pin number inserted	1(B)	X	User prompts to enter pin number	B: Pin number is forwarded for validation
B	Validate pin number	2 (C,D)	A	Pin number provided by the user	C: Valid pin number
					D: Invalid pin number
D	Invalid pin entered	1 (Y)	B	Incorrect pin number message	F: Message displayed for incorrect pin number
C	Valid pin entered	1(E)	B	Correct pin entered message	E: Amount is forwarded for checking
E	Withdrawal amount inserted	2(F, G)	C	User prompts to enter withdrawal amount	F: Withdrawal request not granted
					G: Request granted for valid withdrawal amount
F	Displaying error message	1(Y)	D	Invalid amount entered message	Y: Error message displayed
G	Balance enquiry	1 (Z)	E	Remaining balance after withdrawal operation	Z: Remaining Balance Message printed

Conversion of Sequence Diagram to Sequence Diagram Graph

Sequence diagram explains how the objects are interacting with each other for a particular test scenario. "Fig.3" represents the sequence diagram and sequence diagram graph of withdrawal task of an ATM. Table 2 represents the dependency table for withdrawal operation of ATM which is shown in sequence diagram graph.

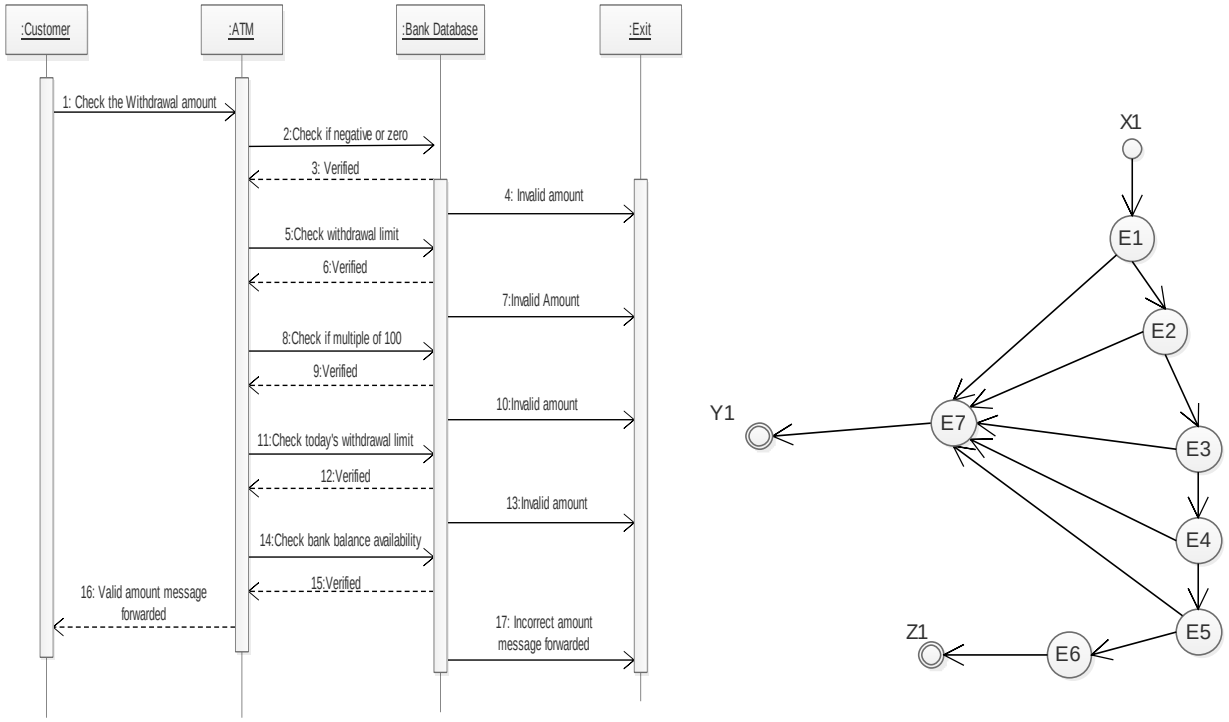

Fig.3. Sequence diagram and sequence diagram graph of withdrawal operation of an ATM

Table 2. Dependency table of withdrawal operation of an ATM through sequence diagram graph

Symbol	Activity Name	Possible number of outputs	Dependency	Input	Expected outputs
E1	Check if amount is non-negative or non-zero	2 (E2,E7)	X1	Amount entered by the user	E2: Amount is forwarded for further checking
					E7: Invalid amount
E2	Check withdrawal limit	2 (E3,E7)	E1	Amount entered by the user	E3: Amount is forwarded for further checking
					E7: Invalid amount
E3	Check if amount is a multiple of 100 or not	2 (E4,E7)	E2	Amount entered by the user	E4: Amount is forwarded for further checking
					E7: Invalid amount
E4	Check today's withdrawal limit	2 (E5,E7)	E3	Amount entered by the user	E5: Amount is forwarded for further checking
					E7: Invalid amount
E5	Check bank balance availability	2 (E6,E7)	E4	Amount entered by the user	E6: Amount is checked
					E7: Invalid amount
E6	Granting withdrawal request	1 (Z1)	E5	Remaining balance after withdrawal operation	Z: Remaining Balance Message printed
E7	Not Granting withdrawal request	1 (Y1)	E1, E2, E3, E4, E5	Invalid amount entered by the user	Y1: Error message displayed

Integration of SCDG and SEDG into SCSEDG

"Fig.4" represents the state-chart sequence diagram system graph (SCSEDG) for the withdrawal operation of an ATM. According to this graph, nodes are connected through edges. Nodes are having one outgoing edge which points to other nodes but some nodes having more than one outgoing edges which are called branches. Nodes, edges, and branches are combined to form the coverage criteria in statechart or sequence diagram graph. Table 3 explains the dependency table of whole ATM operation.

Table 3. Dependency table of Statechart sequence Diagram graph used in ATM Withdrawal Operation

Symbol	Activity Name	Possible number of outputs	Dependency	Input	Expected outputs
A	Pin number inserted	1(B)	X	User prompts to enter pin number	B: Pin number is forwarded for validation
B	Validate pin number	2 (C,D)	A	Pin number provided by the user	C: Valid pin number
					D: Invalid pin number
D	Invalid pin entered	1 (Y)	B	Incorrect pin number message	F: Message displayed for incorrect pin number
C	Valid pin entered	1(E)	B	Correct pin entered message	E: Amount is forwarded for checking
E	Withdrawal amount inserted	1 (E1)	C	User prompts to enter withdrawal amount	E1: Amount is forwarded to be checked
E1	Check if amount is non-negative or non-zero	2 (E2,E7)	E	Amount entered by the user	E2: Amount is forwarded for further checking
					E7: Invalid amount
E2	Check withdrawal limit	2 (E3,E7)	E1	Amount entered by the user	E3: Amount is forwarded for further checking
					E7: Invalid amount
E3	Check if amount is a multiple of 100 or not	2 (E4,E7)	E2	Amount entered by the user	E4: Amount is forwarded for further checking
					E7: Invalid amount
E4	Check today's withdrawal limit	2 (E5,E7)	E3	Amount entered by the user	E5: Amount is forwarded for further checking
					E7: Invalid amount
E5	Check bank balance availability	2 (E6,E7)	E4	Amount entered by the user	E6: Amount is checked
					E7: Invalid amount
E6	Granting withdrawal request	1 (G)	E5	Remaining balance after withdrawal operation	G: Request granted for valid withdrawal amount
E7	Not Granting withdrawal request	1 (F)	E1, E2, E3, E4, E5	Invalid amount entered by the user	F: Withdrawal request not granted
F	Displaying error message	1(Y)	E7	Invalid amount entered message	Y: Error message displayed
G	Balance enquiry	1 (Z)	E6	Remaining balance after withdrawal operation	Z: Remaining Balance Message printed

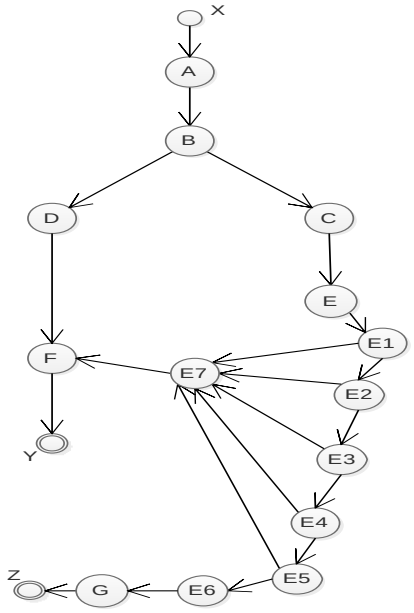

Fig.4. State chart sequence Diagram System Graph of withdrawal operation of an ATM

Integration of SCDG and SEDG into SCSEDG

Algorithm (Conversion of SCDG and SEDG into SCSEDG)

Input: - Statechart Diagram Graph (SCDG) and Sequence Diagram Graph (SEDG)
Output: - Statechart Sequence Diagram Graph (SCSEDG)

1. P= Identify all the possible paths of SCDG
2. For each path p_i, Є P do
3. $S_j = S_i$ // Current state S_j is initialized with initial state S_i
4. T ← φ // T is initialized with a NULL value
5. For each state S_j of the path p_i

   If c_i Є S_i // Check if current state S_i have multiple derived paths

   $\alpha = S_i - 1 \rightarrow$ SEDG // Edge from the previous node to the SEDG

   β = SEDG (last) $\rightarrow S_i + 1$ // Edge from the last node of SCDG to the next node of SCDG

   End If

   T ← T U T1

   If c_i Ȼ S_i

   $\Upsilon = S_i \rightarrow S_i + 1$ // Edge from present node to next node of same SCDG

   End If

6. End For
7. End

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, E1, E2, E3, E4, E5, E6, E7, F, and G are the intermediate nodes describing various operations or activities occurring within the system during operation execution. Out of which A, B, C, D, E, F, and G describes the whole ATM operation without going into the details of withdrawal amount checking. Rest nodes E1 to E7 describe only the withdrawal amount checking operation.

Generation and Optimization of test cases

After creating SCSEDG 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.

PS-BCA (Pseudo code of Test case or Test data generation by using PS-BCA 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 PSBCA to the Activity State Chart Diagram Graph (SCSEDG)
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 the number of 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 Particle Swarm Optimization i.e., Phase 1
Another copy undergoes Bee Colony Optimization i.e., Phase 2

Phase 1
Evaluate different PSO parameters
Update velocity
Update new position with the help of velocity
Fitness Functional value is calculated
Check the boundary conditions
Evaluate its fitness value
If(finess(new)>fitness(old))
then replace the older solution
End If

Phase 2
//Employed Bee Phase
Produce new candidate solution
Check the boundary conditions
Evaluate its fitness value
If(fitness(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
Memorize the best solution
Update Pbest and Gbest values
Generation (k) = Generation(k)+1
Find the best solution so far
End While
Select the best solution having the best fitness functional value
The whole process is repeated until the terminating criterion is made contented.
The best path is found which means that the test cases are optimized.

Phase 1:- (Particle Swarm Algorithm)
The new position of the particle is given by

$$x1=x+v1 \quad (1)$$

Where x=candidate solution
v1= updated velocity value
x1= updated particle position

The velocity for the new solution is given as follows:

$$v1=w*v+c1*(pbest-x)*rand+c2*(gbest-x)*rand \quad (2)$$

Where v1=new solution
rand= a random number in the range of [0,1]
pbest = personal best solution
gbest = global best solution
c1,c2 and w are the PSO parameters

Here weight factor ‘w’ can be calculated as:

$$w= (0.9-(t*0.4)/500) \quad (3)$$

Where t= current iteration number

Phase 2:- (Bee Colony Algorithm)

The new solution can be calculated as

$$c=x(j)+ebf*x(j) \quad (4)$$

Where x(j)= candidate solution at j^{th} position
ebf = a random number in the range of [-1,+1]

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

$$prob(j)=fx(j)/tfx; \quad (5)$$

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

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

$$v(j)=x(j)+ebf*x(jj); \quad (6)$$

Where ebf=a random number is in the range of [-0.1, +0.1]

METHODOLOGY

For Mathematical function

$$f(x)=1/(abs(suc\_bal)+\varepsilon)^2 \quad (7)$$

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

Here Successive Amount (suc_amt) is defined as:

$$suc\_bal = net\_bal - (wtd\_amt - min\_bal) \quad (8)$$

Where net_bal = current account balance
min_bal= minimum bank balance limit

Table 4. Fitness Functional values of each test cases or test data with iteration numbers

	Particle Swarm Algorithm(PSA)		Bee Colony Algorithm(BCA)		Particle Swarm Bee Colony Algorithm (PSBCA)	
Iteration Number	Test cases/Test data	Fitness Function Value	Test cases/Test data	Fitness Function Value	Test cases/Test data	Fitness Function Value
1	4500	6.0966e-010	4000	5.9488e-010	4600	6.1268e-010
10	7100	6.9618e-010	5700	6.4746e-010	8200	7.3841e-010
20	10100	8.2101e-010	7600	7.1491e-010	12200	9.295e-010
30	14100	1.0473e-009	8800	7.631e-010	16200	1.2056e-009
40	19800	1.5747e-009	14600	1.0821e-009	20000	1.6e-009
50	22400	1.9578e-009	19100	1.4907e-009	23300	2.1236e-009
60	24400	2.3565e-009	20900	1.7217e-009	26300	2.8596e-009
70	27400	3.2283e-009	22800	2.029e-009	29300	4.0569e-009
80	31400	5.4065e-009	26600	2.9536e-009	32300	6.1998e-009
90	35300	1.0628e-008	29400	4.1091e-009	35300	1.0628e-008
100	38600	2.4413e-008	32300	6.1999e-009	38300	2.2275e-008
110	41200	6.9248e-008	37000	1.5624e-008	41300	7.304e-008
120	42800	2.0659e-007	39000	2.7777e-008	44000	9.9971e-007
130	43000	2.4997e-007	41700	9.182e-008	44000	9.9971e-007
140	43800	6.9427e-007	43000	2.4997e-007	44000	9.9974e-007
150	43800	6.9432e-007	43700	5.916e-007	44000	9.9979e-007
160	43900	8.2628e-007	44000	9.9974e-007	44000	9.998e-007
170	44000	9.9989e-007	44000	9.998e-007	44000	9.998e-007
180	44000	9.998e-007	44000	9.998e-007	44000	9.998e-007
200	44000	9.998e-007	44000	9.998e-007	44000	9.998e-007

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 Particle Swarm Optimization (PSO) and another copy of candidate solution undergo the second phase i.e., Phase 2 Bee Colony Optimization (BCO).In Phase 1 (PSO), different PSO parameters are calculated. With the help of that parameter, new velocity is obtained and using the velocity new position of the candidate solution found out. If the solution is found to be producing a better solution than the old solution replaced is replaced by the new solution. Phase 2 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 a 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 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. The results gained from both phases are merged. Again all the candidate solutions are sorted and the bottom half worst solution is discarded and is 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 are satisfied. The solution produced so far is the best optimal solution. Table 4 represents the functional value of the fitness function of each test cases or test data with iteration numbers.

V. Simulation Results

The proposed approach generates the automated test cases through test cases or test data for Bank ATM by using PSBCA hybrid algorithms. The "Fig.5." 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 4.

After evaluation, it was found that using Particle Swarm Optimization technique the optimal solution is achieved after 170 iterations whereas by using Bee Colony Algorithm the optimal solution is achieved after 160 iterations. But by implementing the hybrid Bee colony approach (PSBCA) it was observed that the optimal result is achieved much earlier around 120 iterations. The proposed approach generates the test case or test data for Bank ATM's withdrawal operation using Particle swarm, a bee colony and PSBCA algorithm. Table 5 represents the range of fitness functional value with different test cases or test data and also it gives the individual candidate solution according to the fitness functional value range in terms of percentage.

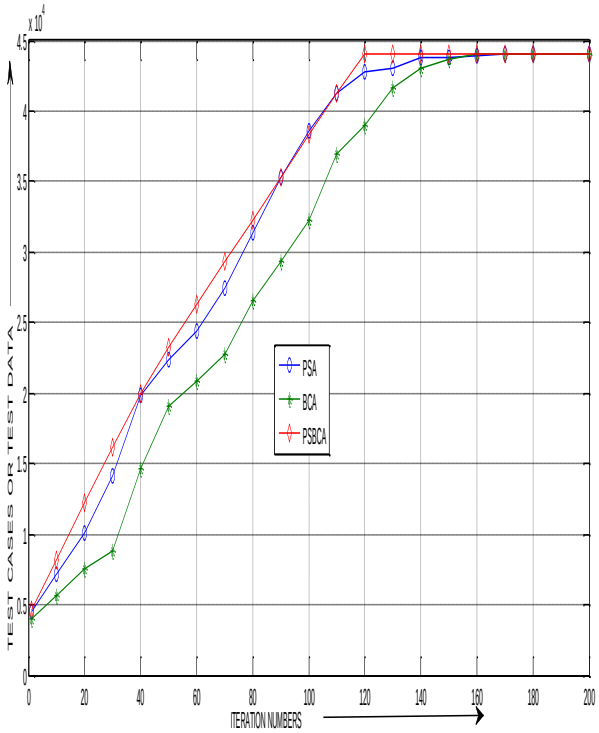

Fig.5. Graphical representation of iteration numbers and test cases or test data values for table 4

Table 5. %of test cases or test data in terms of maximum fitness value

FITNESS VALUE RANGE	% OF TEST CASES/TEST DATA(PSA)	% OF TEST CASES/TEST DATA(BCA)	% OF TEST CASES/TEST DATA(PSBCA)
$0 \leq f(x) < 0.3$	45	40	30
$0.3 \leq f(x) < 0.7$	30	35	25
$0.7 \leq f(x) < 1.0$	25	25	45

The above table shows that around 45% test cases or test data are having the higher fitness function f(x) value and lies in between 0.7 and 1.0 by using hybrid bee colony algorithm (PSBCA) but in case of particle swarm and bee colony algorithm, only 25% of test cases or test data are available within the higher range of fitness function. By considering all the functional value of fitness function from table 4 hybrid bee colony algorithm is having higher fitness value range as compared to particle swarm and bee colony algorithm. "Fig.6" shows a pictorial representation of the relation of two variable quantities like percentage of test cases or test data and fitness value range.

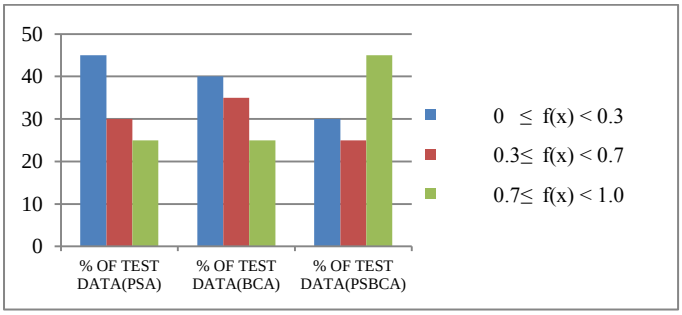

Fig.6. Graphical representation of % test case or test data and fitness value range for table 4

VI. Case Study

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 number of paths that can be traversed through BFS by using SCSEDG system graph. Out of which only one path produces an optimal result and rest six number of paths do not produce the optimal result.

Path 1: X -> A -> B -> D -> F -> Y
Path 2: X -> A -> B -> C -> E -> E1 -> E7 -> F -> Y
Path 3: X -> A -> B -> C -> E -> E1 -> E2 -> E7 -> F -> Y
Path 4: X -> A -> B -> C -> E -> E1 -> E2 -> E3 -> E7 -> F -> Y
Path 5: X -> A -> B -> C -> E -> E1 -> E2 -> E3 -> E4 -> E7 -> F -> Y
Path 6: X -> A -> B -> C -> E -> E1 -> E2 -> E3 -> E4 -> E5 -> E7 -> F -> Y
Path 7: X -> A -> B -> C -> E -> E1 -> E2 -> E3 -> E4 -> E5 -> E6 -> G -> Z

Here Path number 1, 2 3, 4, 5 and 6 all produces incorrect result and can be labeled as Unsuccessful. Only Path 7 produces correct optimized solution and can be regarded as Successful.

Table 6. Path Movement through various nodes in ATM Withdrawal Operation using statechart sequence diagram system 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(m1a,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)
D(m4,b,c)	C(m3,c,b)	C(m3,c,b)	C(m3,c,b)	C(m3,c,b)	C(m3,c,b)	C(m3,c,b)
F(m13,c,d)	E(m5,b,c)	E(m5,b,c)	E(m5,b,c)	E(m5,b,c)	E(m5,b,c)	E(m5,b,c)
State Y >	E1(m5,b,c)	E1(m5,b,c)	E1(m5,b,c)	E1(m5,b,c)	E1(m6,b,c)	E1(m6,b,c)
	E7(m12,b,c)	E2(m6,b,c)	E2(m6,b,c)	E2(m6,b,c)	E2(m7,b,c)	E2(m7,b,c)
	F(m13,c,d)	E7(m12,b,c)	E3(m7,b,c)	E3(m7,b,c)	E3(m8,b,c)	E3(m8,b,c)
	State Y>	F(m13,c,d)	E7(m12,b,c)	E4(m8,b,c)	E4(m9,b,c)	E4(m9,b,c)
		State Y>	F(m13,c,d)	E7(m12,b,c)	E5(m10,b,c)	E5(m10,b,c)
			State Y>	F(m13,c,d)	E7(m12,b,c)	E6(m11,c,b)
				State Y>	F(m13,c,d)	G(m14,b,d)
					State Y>	State Y>

VII. 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 particle swarm, a bee colony and hybrid bee colony algorithm (PSBCA).In particle swarm algorithm the particle is initializing with current position and velocity. The particles keep track of personal best and global best position and update its position and velocity.In this case, the fitness functional value depends on the values of the inputted velocity and variables. It has also been seen that the pbest value depends on upon the gbest value and pbest is directly proportional to the gbest value along with test cases. In Bee Colony Algorithm (BCA) the honey bees are initialized with current position and it also keeps track of best food source position among a number of food source positions. The fitness functional value depends on the input variables and food source position. According to hybrid PSBCA algorithm, the test cases are optimized and various paths are generated through UML diagrams like state chart and sequence diagrams. Then the combinational system graph like SCSEDG (combination of state chart graph and sequence diagram graphs) is designed. This paper also explains the traversing of the graph through DFS which is implemented in MATLAB 7.The future approach to this work could enhance the test case or test data generation for large Programs automatically. In future different hybrid search, based optimization technique like BBCA, GFA, and BCBA may be used to generate the test cases from combinational diagrams with path coverage which also improve the software quality. The test cases or test data generated by using PSBCA is compared with test cases or test data generated by PSA and BCA and it was found that PSBCA produces an optimal result in very less time and with more accuracy.

VIII. Conclusion

Testing ensures the necessary condition of software, Generation, and optimized test cases have to be addressed software testing. In software testing factors like cost, time and effort are influenced. The objective of this proposed Hybrid bee colony algorithm is to optimization of test cases, generation of path coverage within minimal execution time and it gives the better result in comparison with particle swarm and Bee colony algorithm. The proposed system takes less time to choose the best test path and it is more capable, reliable for the development of software.

REFERENCES

[1] A.A. Kyaw and M.M. Min, "Model -Based automatic optimal test path generation via Search optimization techniques; A critical review, "In Proc. the 12th International conference on computer Applications, 2014.

[2] A.V.K.Shanthil and G.Mohan Kumar "Automated Test cases generation from UML sequence diagram", 2012, International conference on software and computer applications (ICSCA.2012) IPCSIT Vol.41(2012) copyright (2012) IACSIT Press, Singapore.

[3] Abdurazik, A and Offutt, J. 2000 Using UML collaboration diagrams for static checking and test generation, proceedings of 3rd international conference UML, lecture notes in Computer Science,2000, PP,383-395.

[4] D. Dervis Karaboga, An Idea Based On Honey Bee Swarm for Numerical Optimization, Technical Report-TR06, Erciyes University, Engineering Faculty, Computer Engineering Department 2005.

[5] 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, Indianapolis (2006)

[6] J. Kennedy, R. C. Eberhart, "Particle swarm optimization", IEEE International Conference on Neural Networks, Piscataway, NJ., pp.942-1948, 1995.

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

[8] Korel, B.1990.Automated software test data generation, IEEE, Trans. Software Engineering, 16(8), 1990, pp.870-879.

[9] Rajesh Kumar Sahoo, Deeptimanta Ojha, Satyabrata Das,"Nature-Inspired Metaheuristic Algorithms-A Comparative Review", International Journal of Development Research,vol.6, Issue-07,pp.8427-8432,2016, ISSN:2230-9926.

[10] Jones, B.F., Department of Computer studies, Glamorgan University, Ponty pride, U.K, Sthamer, H-H, Eyres, D.E., Automatic structural testing using Genetic algorithms, Software Engineering Journal, Volume 11, Issue-5, September 1996, pp. 299-306.

[11] N. Narmada and D.P.Mohapatra, "Automatic test data generation for data flow testing using particle swarm optimization ", Communications in computer and information Science, Vol.95, No-1, pp. 1-12,2010.

[12] L.T.Bui,et al. "A modified strategy for the construction factor in particle swarm optimization," in book series lecture Notes in computer science Vol 4828/2010, ed.Heidelberg: Springer Berlin, 2010, pp. 333-344.

[13] 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.

[14] 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.

[15] Supapornkansomkeat and WanchaiRivepiboon, "Automated-Generating test case using UML state chart Diagrams" Proceedings of SAICSIT 2003.

[16] 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.

[17] 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 July 2010.

[18] S. Singla, D.Kumar, H.M. Rai, P.Singla, A hybrid PSO approach to automating Test data generation for data flow coverage with dominance concepts, International journal of advanced science and technology, Vol.37.

[19] 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,pp.54-58,ISSN:2347-2693,2016.

[20] S .S. Priya and P.D.Sheba. "Test case Generation from UML models- A Survey, " In Proc. International conference on information systems and computing(IC ISC-2013). India, January 2013, Vol.3, Special issue.1.

[21] Philip Samuel, Rajib Mall and Sandeep Sahoo "UML sequence diagram based testing using slicing", IEEE indiction 2005 conference, Chennai, India,11-13 Dec 2005, pp. 176-178.

[22] Rajesh Kumar Sahoo, Deeptimanta Ojha, Durga Prasad Mohapatra, Manas Ranjan Patra,"Automatic generation and optimization of test data using harmony search algorithm", ACITY,pp.23-32,2016

[23] S P Tripathy, P M Sahu, D Kandahar. Optimization of Discrete and continuous test suite using Heuristic Algorithm: A Comparative study, IEEE conference on information and communication Technologies (ICT 2013), pp. 841-846 April 2013.

[24] Rajesh Kumar Sahoo, Deeptimanta Ojha, Durga Prasad Mohapatra, Manas Ranjan Patra,"Automatic generation and optimization of course timetable using a hybrid approach","Journal of Theoretical and Applied Information Technology, Vol.95, No.1,pp.68-77,2017

Authors' Profiles

Rajesh Ku. Sahoo received his Master's degree from KIIT University Bhubaneswar. He is presently working as assistant professor in computer science & engineering department of ABIT Cuttack. His area of interest is software engineering, software testing, and computer architecture. He has contributed research papers in various international journals and conferences. He is having fifteen years of teaching experience. He is a member of ISTE.

Dr. Santosh Kumar Nanda is working as Professor in Eastern Academy of Science and Technology, Bhubaneswar, Orissa, India. He completed his Ph.D. from National Institute of Technology, Rourkela. His research interests are Soft Computing, Artificial Intelligence, Image Processing, Prediction of machinery noise and vibration, Noise and vibration control, Mathematical modeling, Pattern Recognition. He has more than 60 research articles in reputed International Journals and International conferences etc. He is also serving more than 15 journals as Editor/Reviewer. So far,

Dr. Nanda has produced 1 Ph.D. and more than 15 M.Tech Scholars. Now-a-days he is supervising another 1 Ph.D. and 6 M.Tech students, respectively. He is also International Program Committee Member of 18th Online World Conference on Soft Computing in Industrial Applications (WSC18) 2014, Committee Member of 17th Online World Conference on Soft Computing in Industrial Applications (WSC17) 2012,16th Online World Conference on Soft Computing in Industrial Applications (WSC16) 2011 and 15th Online World Conference on Soft Computing in Industrial Applications (WSC15) 2010. He is now selected as International Program Committee of 2014 and 2013 World Conference on Information Systems and Technologies (WorldCIST, France). Currently, he is an Individual Member of International Rough Set Society and Member of International Association of Engineers (IAENG). In 2014, he was selected as a Young Researcher Member in World federation of Soft Computing, USA. Dr. Nanda is now acting as Associate Editor of Prestigious Journal International Journal of Intelligent System, Hong Kong and Editor in Chief of International Journal of Engineering and Manufacturing (IJEM), Hongkong. He is also acting as Editor in Chief of Journal Artificial Intelligence, USA.

Dr. Durga Pr. Mohapatra received his Master's degree from National Institute of Technology, Rourkela, India. He has received his Ph.D. from Indian Institute of Technology, Kharagpur, India. He is currently working as an Associate Professor at National Institute of Technology, Rourkela. His special fields of interest include Software Engineering, Software testing, Discrete Mathematical Structure, Program Slicing and Distributed Computing. Many publications are there to his credit in many International and National level journal and proceedings.

Dr.Manas Rn. Patra has received his Ph.D. from Central University, Hyderabad, India. He is currently working as the reader in computer science and engineering department, Berhampur University, Berhampur, India. His areas of interests are Intelligent Agents, Service Oriented System Modeling, Data mining, and Network Intrusion Detection. He is contributed many research papers in various international journals and conferences.

How to cite this paper: Rajesh Ku. Sahoo, Santosh Kumar Nanda, Durga Prasad Mohapatra, Manas Ranjan Patra,"Model Driven Test Case Optimization of UML Combinational Diagrams Using Hybrid Bee Colony Algorithm", International Journal of Intelligent Systems and Applications(IJISA), Vol.9, No.6, pp.43-54, 2017. DOI: 10.5815/ijisa.2017.06.05

