International Journal of Applied Metaheuristic Computing
Volume 13 • Issue 1

Forest Optimization-Based Test Case Generation for Multiple Paths With Metamorphic Relations

Rashmi Rekha Sahoo, Siksha 'O' Anusandhan (Deemed), India*
Mitrabinda Ray, Siksha 'O' Anusandhan (Deemed), India

ABSTRACT

In general, multiple paths are covered by multiple runs, which is a time-consuming task. Metaheuristic techniques are widely used for path coverage. In order to reduce the time, an efficient method is proposed based on forest optimization algorithm (FOA) with metamorphic relations (MRs) that cover multiple paths at a time in one run, unlike the traditional search-based testing. In the proposed approach, initial test case is generated using FOA; the successive test cases are generated using MRs without undergoing several runs. The motive of using FOA is that the searching mechanism of this algorithm has resemblance with the branch/path coverage techniques of testing. To the best of the authors' knowledge, FOA has not been implemented in software testing. The experimental results are compared with three existing works. The efficiency of simply FOA is also shown to cover multiple paths. The results show that FOA with MRs is more efficient in terms of time consumption and number of paths covered.

KEYWORDS

Branch Coverage, Forest Optimization, Metaheuristic Techniques, Metamorphic Relations, Multiple Path Coverage, Path Coverage, Software Testing, Test Case Generation

1. INTRODUCTION

Testing is a very labour intensive task and consumes 40-70% of time and resources of software development process (Khana et al 2019). Test case generation is best fitted to a multi objective problem. Because, the desired outcome of test case generation activity is maximum coverage in minimum time. Various searching techniques (Yao & Gong, 2014; Khana et al., 2018) are designed to deal with multi objective problem. Application of searching techniques in software testing is called Search Based Software Testing (SBST) (Harman et al., 2015). Though, many meta heuristic algorithms like Genetic Algorithm (GA), Particle Swarm Optimization (PSO), Cuckoo Search (CS), Ant Colony Optimization (ACO) etc. (Sahoo & Ray, 2018) are applied till now in automatic test case generation, still there is a need for reduction in execution time. Most of them are used to generate test case for one path at a time which is time consuming (Manikumar et al., 2016).

Main objective of this paper is to cover multiple target paths within the available test resources at the time of automatic test case generation. Improvement in the search based testing is done by

DOI: 10.4018/IJAMC.292503

*Corresponding Author

adding metamorphic relations (Chen, 1998) between test cases, in order to reduce the number of runs of meta heuristic algorithm, in path coverage. In this paper, metamorphic relations among a set of test cases are used to generate next set of test cases. Metamorphic testing tests whether the selected inputs to a programme under test and their outputs satisfy certain relations. The relationship between inputs and outputs is termed as Metamorphic Relation (MR). After multiple programme executions, if any of the MRs are not satisfying in any execution, then at least one error exists in the programme and the next test cases cannot be generated (Chen et al,. 1998; Kanewala et al., 2013). In this paper, FOA (Ghaemi & Feizi-Derakhshi, 2014) is used to generate a set of test cases. The first set of test cases (successful) are generated through FOA, the test cases for rest of the paths are generated using metamorphic relations in the same run without undergoing several runs. It takes less time and less iterations. As far best of our knowledge, this is the first work in FOA for test case generation. FOA is inspired by few trees in the forest which can survive for several decades. In FOA, seeding procedure of the trees is simulated. FOA is being chosen because candidate solutions are generated exponentially yielding a faster search. It provides a promising result in different areas (Sahargahi & Derakhshi, 2017; Chaghari et al, 2018; Mohanty et al., 2018). It is a suitable algorithm for test case generation based upon branch coverage and path coverage in software testing. In this paper, Branch distanced based fitness function combined with approximation level (Baresel et al., 2002; Chen et al, 2009) and Improved Combined Fitness (ICF) (Sahoo & Ray, 2020) are applied. The objective function is designed based upon the path coverage criteria. Through experiments, the performance of FOA in test case generation for multiple target paths is evaluated.

Research gaps identified in the existing literatures are as follows:

- In the existing literatures (Lakshminarayana, P., & SureshKumar, T. V. 2020; Baresel, A., Sthamer, H., & Schmidt, M.; 2002), they focus on covering a single target path in one run which is time consuming task. Our proposed approach aims at multipath coverage.
- Few work (Fangqing Liu et al.,2019; Panichella et al., 2018; Ziming Zhu et al.; 2017; Yao & Gong, 2014) has been done on multipath coverage either by improving metaheuristic algorithms or by improving fitness function to reduce the number of execution of the algorithm. It solely depends on the performance of the meta heuristic algorithm and the fitness function. The proposed approach use metamorphic relation to avoid several execution of meta heuristic algorithm.
- Lv et al., (2018) avoid several execution of metaheuristic algorithm by using metamorphic relations. Initial test case is generated using PSO. In PSO, at a time, one candidate solution is generated from a solution which makes the convergence process slower. Hence, it is required to select appropriate searching techniques. In FOA, candidate solutions are generated exponentially by the parameter LSC which generates more than one solution from a single solution. It increases the probability of covering more than one path (true and false branch) in one iteration unlike GA, PSO, ABC, CS etc.

This paper has two contributions: first, proposal of FOA for test case generation. Second proposal is the combination of metamorphic relation with FOA for test case generation that covers multiple paths in one run.

The remaining part of this paper is organized as follows: In Section 2, background study of FOA, metamorphic relations. In Section 3, related work on FOA and multipath test case generation have been discussed and a clustered view of work is presented. Section 4 presents the proposed method. In Section 5, Methods and materials used are shown. Experimental results of case study and its comparison with existing work is discussed in section 6. Section 7 represents threats to validity. Conclusion and future work are presented in Section 8.

2. BACKGROUND

In this section, FOA and metamorphic relations are discussed.

2.1 Forest Optimization Algorithm (FOA)

FOA proposed by Manizheh Ghaemi and Mohammad-RezaFeizi-Derakhshi (Ghaemi & Feizi-Derakhshi, 2014). This algorithm is inspired by the trees which survive for many decades. The objective of the algorithm is to search for a tree (candidate solution) in the forest (set of candidate solutions) which survives for a long time. It simulates the seeding procedure of nature in a forest. Some seeds fall near the parent tree (called local seeding), incorporates the local search process. Dispersal of seeds far away from the parent tree, incorporates global searching. FOA has three stages: local seeding, population limiting and global stage. The flow chart of FOA shown in Figure1 is described below.

Initially, the forest is assigned with an initial population. Each tree has 'n' variables and has a part called 'age' and it is initialized with zero. In local seeding phase, from each zero aged tree, a fixed number of trees are generated by adding a small value to few variables of a tree, decided by Local Seeding Change (LSC). In each iteration, some new trees are generated and their age is set to zero. The old trees are getting older by 1.

In population limiting stage, population is controlled by two parameters that are 'life time' and 'area limit'. Age limit of a tree is set through the parameter life time. When the tree crosses its age limit, it is added to candidate solution. Forest is set with a population called 'area limit'. Extra trees exceeds from the area limit are added to the candidate solution from which seeds are chosen for global seeding.

In global seeding, number of seeds will be transferred to distance area is decided by Transfer Rate (TR). Global Seeding Changes (GSC) parameter is set to decide how many variables value will be changed in a tree.

After global seeding, best tree is updated according to fitness value and its age is reinitialized to zero to avoid aging problem. After that best tree is removed from the forest. These stages are repeated iteratively as long as the condition is not met.

2.2 Metamorphic Testing and Metamorphic Relations (MRs)

Metamorphic testing is introduced to overcome the oracle problem (Chen et al,. 1998). Oracle problem occurs when the expected output of an input is not present for the software under test (SUT). In this situation, one way is to check whether both the input and the actual output satisfy a particular relation or not. Such relation between input and output is called metamorphic relation and the testing done using this relation is called metamorphic testing. If any of the MRs is not satisfying while running the SUT, then there exists at least one error in the programme (Kanewala et al., 2013). MR is explained below.

Suppose x1, x2,…., xn are given input data of function 'f' of programme 'P', having a relation 'ir' between them called Input Relation(IR), and f (x1), f (x2), …, f (xn) are the corresponding outputs having a relation 'or' between them called Output Relation(OR), i.e.,

$$ir(x1, x2, \ldots, xn) \Rightarrow or\ (f\ (x1), f\ (x2), \ldots, f\ (xn)) \quad (1)$$

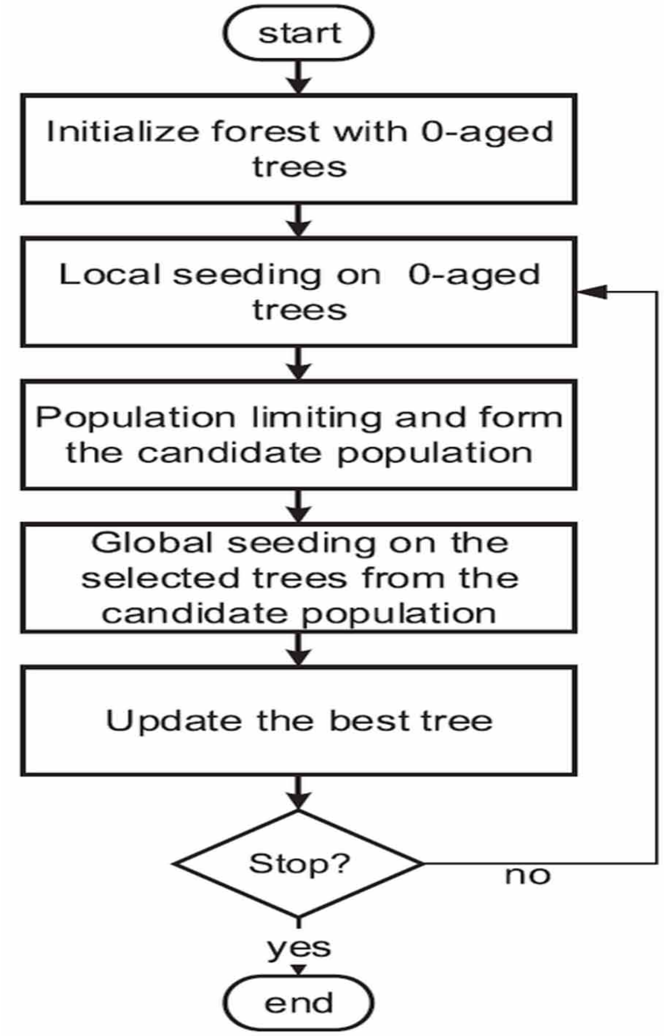
(ir, or) is called the Metamorphic Relation(Lv et al., 2018). If P is correct, then it must satisfy:

$$ir\ (I1, I2, \ldots, In) \Rightarrow or\ (P(I1), P(I2), \ldots, P(In) \quad (2)$$

I1, I2, …, In are actual inputs of P and PI1, PI2, …, PIn are the corresponding outputs.

In metamorphic testing, input test cases are designed in two phases. In first phase, the input test cases called origin test case (OTC) are derived from other testing techniques and in second phase, input test cases are called follow-up test case (FTC), which are designed based on OTC and IR. For instance, for a programme that implements function sin(x), there is a MR called MR1: $\sin(x) = \sin(x + 2\pi)$, x is called OTC, and $x + 2\pi$ is called FTC.

Figure 1. Flow chart of FOA (Ghaemi & Feizi-Derakhshi, 2014)


start
Initialize forest with 0-aged trees
Local seeding on 0-aged trees
Population limiting and form the candidate population
Global seeding on the selected trees from the candidate population
Update the best tree
Stop?
no
yes
end

Now, the question is how we can use MRs for multi path coverage. As OTC and FTC have different values that may cover different paths. In sin(x) function, the path covered by OTC (x) is different from FTC (x + 2π). The effectiveness and applicability of metamorphic testing is determined by MRs. Therefore, MRs design is a very important aspect. In this paper, constructions of MRs have not been focused because it needs expertise. Already constructed and tested MRs used in other research papers ((Lv et al., 2018; Chen et al., 1998) are used in this paper.

2.3 Triangle Programme

This is a triangle area computing programme. It takes three sides as input and finds the type (scalene, isosceles, equilateral and invalid triangle) of triangle and compute the area of that triangle. There are six paths computed using Cyclomatic Complexity (CC) (Mall, 2018). Figure 2. shows MATLAB code of Triangle area computation programme and Figure 3. shows its corresponding Control Flow Graph(CFG).

3. RELATED WORK

FOA is a limited used meta heuristic algorithm till now. Few work (Ghaemi & Feizi-Derakhshi, 2016; Sahargahi & Derakhshi, 2017; Chaghari et al, 2018) have been done applying this algorithm.

In section 3.1, literature review of some applications of FOA in different areas is provided and also the application of other searching techniques (GA, PSO etc.) in test case generation for multipath is discussed in section 3.2.

3.1 Implementation Area of FOA

Manizheh Ghaemi and Mohammad-RezaFeizi-Derakhshi(2014) propose Feature Selection using Forest Optimization Algorithm (FSFOA). Experimental results show classification accuracy of classifiers is using FSFOA is more than PSO. Arash Chaghari et al. (2018) use FOA and combine it with gradient method for fuzzy clustering. Accuracy of the proposed method is compared with fuzzy clustering based on PSO and GA. It shows that fuzzy clustering based on FOA gives better accuracy than other two. Subhadarshini Mohanty et al. (2018) use FOA for load balancing in cloud computing. They find the distinguished solutions from the pool to avoid starvation of tasks as well as improve the average response time. Shadi Mahmoudi and Chia Lailypour (2015) propose Binary Forest Optimization Algorithm (BFOA). Result shows BFOA outperforms Binary GA, PSO, and CS. R Jaya Suji and SP Rajagopalan (2016) propose a feature selection algorithm called multi-ranked feature selection algorithm for effective feature selection to be used in a new cancer detection system. Xie et al. (2020) propose improved FOA using pearson correlation coefficient and L1 regularization techniques to improve the classification accuracy. They use to generate co-related feature set to avoid random initialization of feature set.

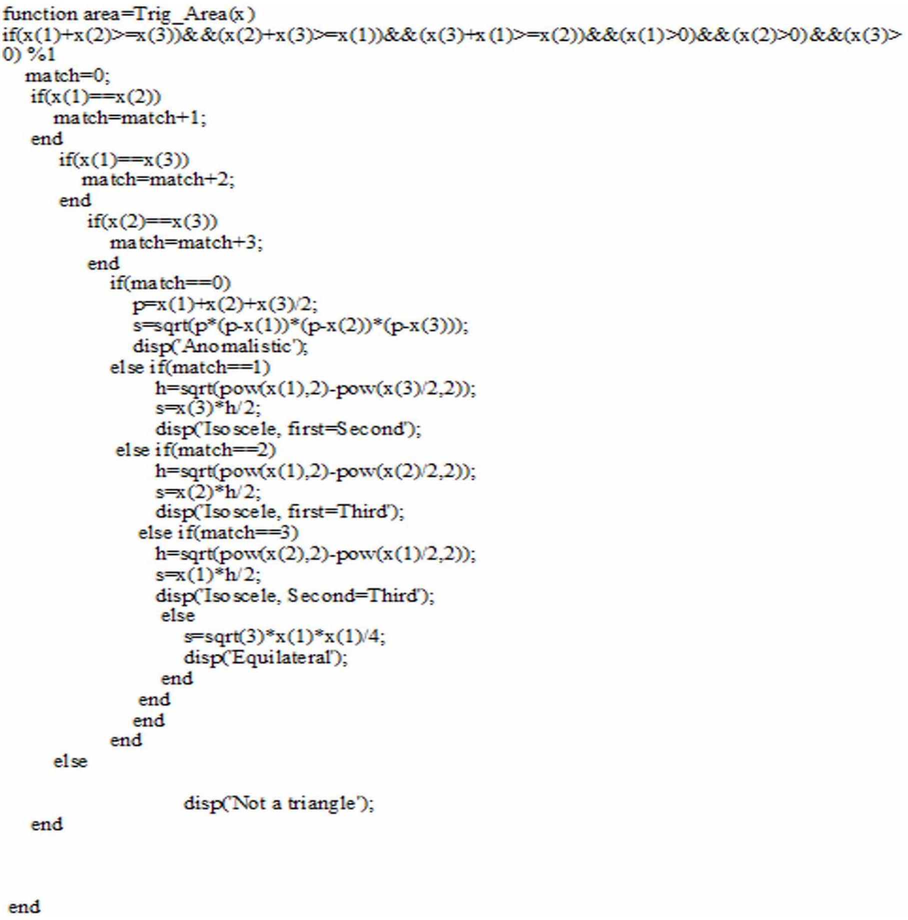
FOA is applied mostly in data mining techniques such as feature selection, reduction, clustering and classification. Next, we discuss some approaches that propose to cover multiple paths.

3.2 Multi Path Test Case Generation Using Meta Heuristic Techniques

Arcuri and Fraser (2014) propose Whole Suite (WS) generation approach using in which they generate all the test cases at a time instead of evolving one by one with in the testing budget. Annibale Panichella et al. (2018) cover many target paths at a time by proposing many objective solver, Dynamic Many-Objective Sorting Algorithm (DynaMOSA). DynaMOSA outperforms WSA and MOSA for statement, branch and mutation coverage but time consumption and memory consumption have not been taken into consideration while covering so many targets at a time. Lv et al. (2018) propose a novel approach to reduce the time consumption by combining PSO algorithm with metamorphic relation to generate test cases for multiple paths. Their aim is to cover multiple paths in one run instead of several runs which is time taking. In intial phase, PSO takes more time to generate OTC. Zhu et al. (2017) propose a GA based approach to reduce the search space of test data by improved grouping strategy to cover multiple paths. Convergence speed becomes faster and the efficiency of searching process is improved. Fangqing Liu et al. (2019) propose a scatter search strategy combined with GA, PSO and Differential Evolution (DE)) to cover all possible paths of Natural Language Processing (NLP) programs. Their test results show that the proposed approach consumes less running time to cover the paths. Ahmed and Hermadi et al. (2008) propose a GA based novel approach for multiple-path coverage. To guide the search process, average of the fitness values of all target paths is considered. Experimental results show it is an efficient method for multipath coverage. Yao and Gong et al. (2014) propose a population based GA for many path coverage. Test case generated from each group covers a single path. They use the individual sharing concept to find the optimal solution which enhances the efficiency of the approach.

Table 1 gives the clustered view and comparison study of the related work and the proposed work. Related work is segregated into two groups: application of FOA in different areas and application of meta heuristic algorithms in multipath test case generation.

Figure 2. Source code of Triangle Programme



```
function area=Trig_Area(x)
if(x(1)+x(2)>=x(3))&&(x(2)+x(3)>=x(1))&&(x(3)+x(1)>=x(2))&&(x(1)>0)&&(x(2)>0)&&(x(3)>
0) %1
   match=0;
    if(x(1)==x(2))
       match=match+1;
    end
        if(x(1)==x(3))
           match=match+2;
        end
            if(x(2)==x(3))
               match=match+3;
            end
               if(match==0)
                  p=x(1)+x(2)+x(3)/2;
                  s=sqrt(p*(p-x(1))*(p-x(2))*(p-x(3)));
                  disp('Anomalistic');
               else if(match==1)
                     h=sqrt(pow(x(1),2)-pow(x(3)/2,2));
                     s=x(3)*h/2;
                     disp('Isoscele, first=Second');
                else if(match==2)
                     h=sqrt(pow(x(1),2)-pow(x(2)/2,2));
                     s=x(2)*h/2;
                     disp('Isoscele, first=Third');
                  else if(match==3)
                     h=sqrt(pow(x(2),2)-pow(x(1)/2,2));
                     s=x(1)*h/2;
                     disp('Isoscele, Second=Third');
                      else
                         s=sqrt(3)*x(1)*x(1)/4;
                         disp('Equilateral');
                      end
                  end
                 end
               end
      else

                         disp('Not a triangle');
    end


 end
```

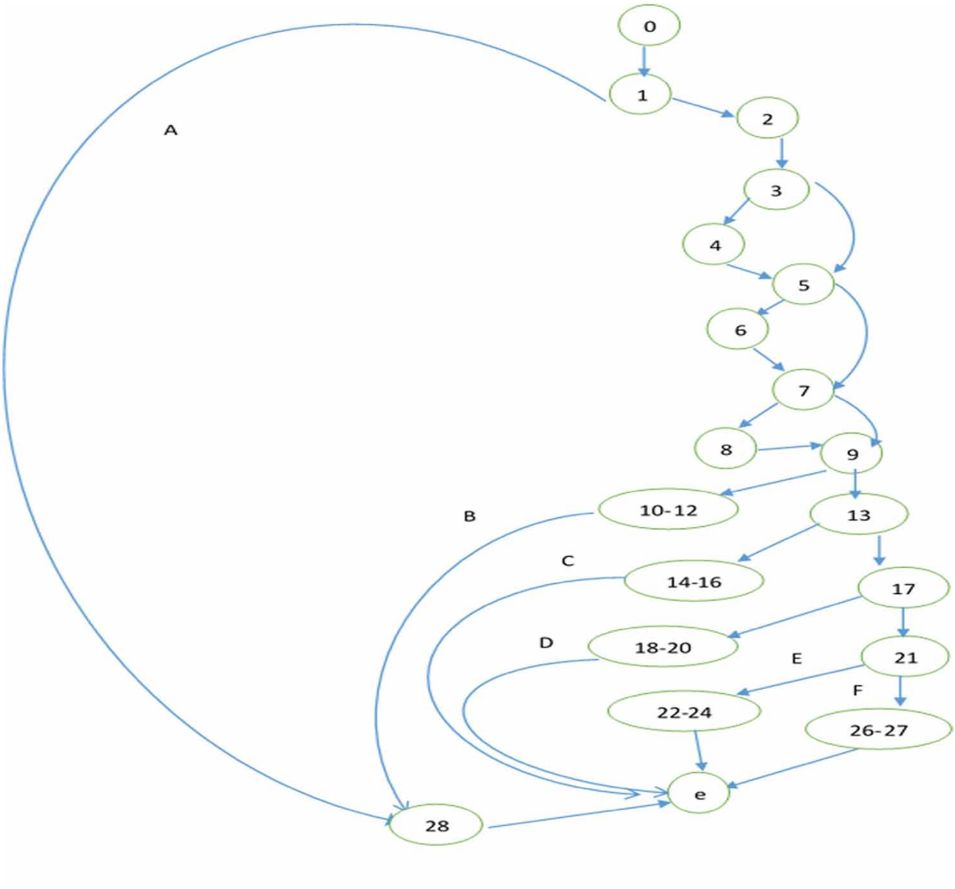
Above discussed multipath coverage methods aim at executing the metaheuristic algorithm as less time as possible to reduce the time consumption. In this paper, proposed FOA generates initial test case faster than PSO and generates successive test cases using metamorphic relation to reduce the execution time for test case generation. It is represented in next section.

4. PROPOSED APPROACH ON TEST CASE GENERATION

4.1 Multipath Test Case Generation Using FOA and MRS

Test cases are generated in two phases as shown in Figure 4. In first phase, one test case is generated using FOA that is the input to MR based test case generation. Instead of generating random test data we generate effective test case as OTC to generate FTC in the second phase. If the target paths are not achieved, FTC becomes OTC and generates follow up test cases and covers the target. If some target paths are not covered by the metamorphic based test cases then only FOA is executed to generate OTC to cover the paths. The previous OTC is discarded. This process goes on till all the targeted paths are covered.

Figure 3. CFG of Triangle Programme



4.2 FO Based Test Case Generation

Figure 5. shows the step by step approach of FOA based test case generation.

The best test case generated using FOA, is treated as OTC and is the input to the MR based test case generation. To cover all the paths, multiple execution of FOA is not required whenever the targets cannot be covered by the FTC then only the FOA is executed. From SUT, CFG is generated. Using CC linearly independent paths are determined. Branch distance functions are used in FOA to generate test data.

4.3 MR Based Test Case Generation

The best test case generated by FOA is treated as OTC for MR-based test case generation process.

4.3.1 Algorithm

```
Input: CFG, Target path sets, MRs
Output: Ordered test cases
```

1. Initialize Covered Path Set(CPS)={}, Test Case Set(TCS)={}, Target Path Sets(TPS)={T1,T2,T3,.., Tn}-Totc, where n is number of linearly independent paths in the given CFG, optimized test suite(OTS)={}

Table 1. Clustered view of literature survey

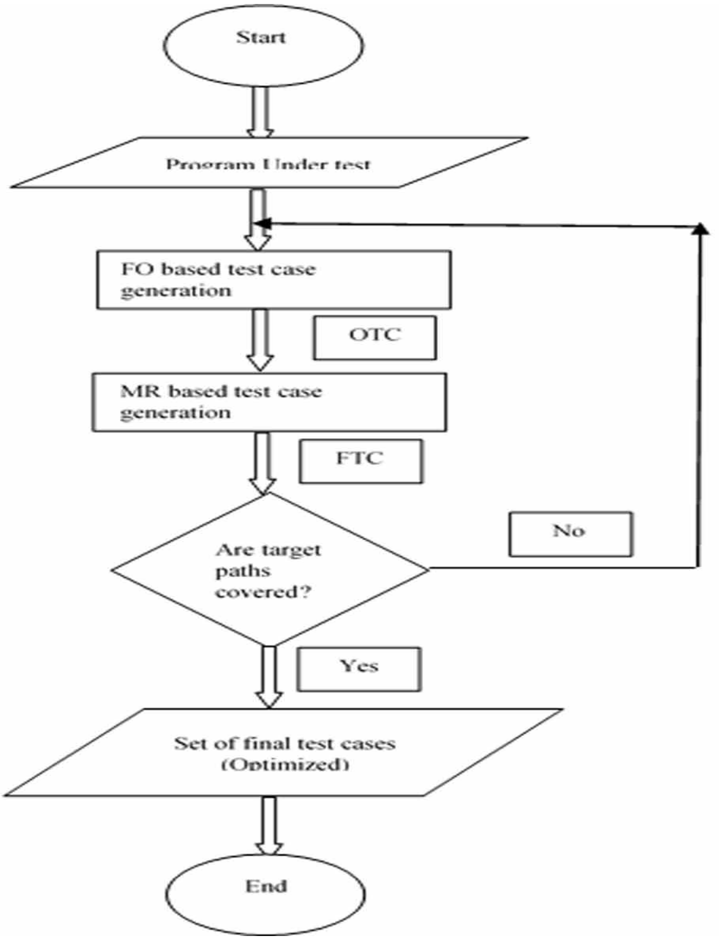
Criteria	Author (Year)	Objective	Searching Technique	Findings	Remark
Applications of FOA	Xie et al.,2020; Manizheh Ghaemi and Mohammad-RezaFeizi-Derakhshi(2014), R Jaya Suji and SP Rajagopalan (2016)	feature Selection	FOA, Improved FOA	improve the classification accuracy of classifiers	FOA outperforms GA, PSO in different application areas. It is not applied in test case generation till now to best of our knowledge.
	Arash Chaghari et al. (2018)	fuzzy clustering	FOA	FO gives better accuracy than PSO and GA	
	Subhadarshini Mohanty et al. (2018)	load balancing in cloud computing	FOA	improve the average response time	
	Shadi Mahmoudi and Chia Lailypour (2015)	Propose modified FO for discrete and binary problem	FOA	Faster convergence like continues FOA	

Applications of meta heuristic techniques in multi Path test case generation	Fangqing Liu et al. (2019);
	Panichella et al. (2018), ;
	Ziming Zhu et al. (2017);
	Arcuri & Fraser (2014);
	Ahmed and Hermadi (2008);
	Yao and Gong (2014)
		To cover multiple paths	GA	Out performs the existing approaches, difficult to cover all the paths and consumes more resources	Though their proposed methods outperforms the methods they compared, Still there is a need of execution time reduction by reducing the number of execution of the metaheuristic algorithm.

	Lv et al. (2018),	To cover multipath using metamorphic relation	PSO	Better success rate and less average no. of evaluations required	Though it requires less fitness evaluation in comparison to other techniques, still the initial phase (test case generation using PSO) takes more time which can be further be reduced using appropriate meta heuristic algorithm.
Our proposed work (Application of FOA for multipath coverage using MRs)		To cover multiple path in single run using metamorphic relation	FOA	Outperforms PSO with metamorphic relation.	Initial phase takes less time as FOA convergence faster and MRs avoids frequent execution of the algorithm. Hence, the overall execution time reduced to a significant extent.

2. Apply mri Є MRs to generate FTC from OTC generated by FOA, where i=1…n
3. Add new FTC in TCS. Update TCS. TCS={FTCi}, where i=1..n
4. Repeat steps 4 and 5 till all the mr_i used to generate FTCs.
5. Select FTCi from TCS.
6. Execute FTCi on program under test. Determine the target path covered.
7. Remove the covered target path from TPS and add in CPS. Update TPS and CPS
8. Repeat steps 8 and 9 till TPS is not empty otherwise go to step 12
9. Obtain the final OTS.
10. Terminate the algorithm

Figure 4. Schematic Diagram of Multipath test case generation using FOA and MRs


Start
Program Under test
FO based test case generation
OTC
MR based test case generation
FTC
Are target paths covered?
No
Yes
Set of final test cases (Optimized)
End

4.4 Implementation of the Algorithm for Test Case Generation

4.4.1 Case Study: Triangle Area Computing Problem

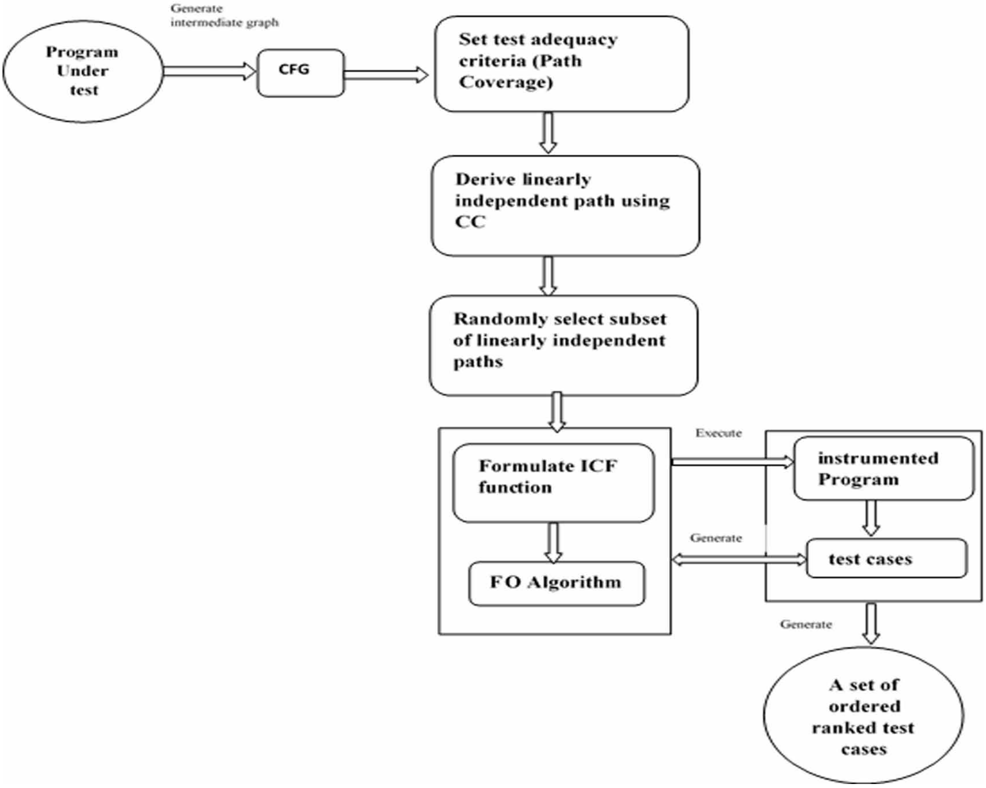
Triangle area computing problem (See Figure 2.) is taken as case study to explain the proposed approach. Path D, E, F shown in Figure 3. are the target paths. MRs used for this programme are, MR1 and MR5, given in Section 5. MR based Test case generation algorithm describes the steps to generate test cases to cover the multiple paths.

Target Path Set (TPS) = {4, 5, 6}, Covered Path Set (CPS) = { }, Test Case Set= { }, Result Set= { }

Step 1. Suppose the best test case generated (5, 5, 5) using FOA covers the target path F (See Figure 3. This is considered as OTC.

Step2. Path F is removed from TPS and added in CPS and in the result set.

Figure 5. FO based Test case Generation


Generate intermediate graph
Program Under test
CFG
Set test adequacy criteria (Path Coverage)
Derive linearly independent path using CC
Randomly select subset of linearly independent paths
Formulate ICF function
FO Algorithm
Execute
instrumented Program
Generate
test cases
Generate
A set of ordered ranked test cases

Step3. Now with MR based test case generation the FTCs are generated. This step is repeated till the TPS is empty. Applying MR1 and MR5 on OTC (5, 5, 5), test case (5, 5, 5) and (5, 7, 5) are obtained as FTC

Step 4. (5, 7, 5) covers the target path D and it is treated as OTC. Path D is removed from TPS and added in CPS. Again MR1 and MR5 are applied on (5, 5, 5) and the obtained FTCs are (7, 5, 5) and (5, 6, 5). (7, 5, 5) cover the target path E. Path 5 is removed from TPS.

Now, the TPS is empty means all the target paths are covered and the test case is added in the result set. So the final test cases are {(5, 5, 5), (5, 7, 5), (7, 5, 5)}.

5. MATERIALS AND METHODS

Algorithm implementation is done in MATLAB. Six subject programs are taken for evaluating the performance. Table 2 shows the parameter setting for FOA. Parameter settings follow a standard values which are mostly used many research papers (Ghaemi & Feizi-Derakhshi, 2014; Sahargahi & Derakhshi, 2017; Chaghari et al, 2018). Table 3 shows description of the subject programs in terms structures used in the program and number of target paths are selected to be covered. For MR based test case generation the MRs are used, defined in other research papers (Chen et al., 1998; Dong et al., 2007). The MRs are defined for each program. The solution domain range for every program taken in this paper is set as [1-1000] except for sin(x) programme. Its range is [-1000 - +1000].

5.1 MRs for the Programme Under Test

Metamorphic relations of 6 subject programmes (Lv et al., 2018) are given in this section

5.2 MR Construction

The triangle area computing program has six MRs. (a, b, c) represents the three sides of the triangle and taken as OTC to generate FTC (a', b', c'). TriSquare(a, b, c) represents the area of the triangle.

MR1: If (a', b', c') = (b, a, c), then
TriSquare(a', b', c') = TriSquare(a, b, c) .
MR2: If (a', b', c') = (c, b, a), then
TriSquare(a', b', c') = TriSquare(a, b, c) .
MR3: If (a', b', c') = (a, c, b), then
TriSquare(a', b', c') = TriSquare(a, b, c) .
MR4: If (a′, b′, c′) = (a, b,), then
TriSquare(a′, b′, c′) = TriSquare(a, b, c) .
MR5: If (a′, b′, c′) = (a, $\sqrt{2a^2+2c^2-b^2}$, c), then
TriSquare(a′, b′, c′) = TriSquare(a, b, c) .
MR6: If (a′, b′, c′) = ($\sqrt{2b^2+2c^2-a^2}$, b, c), then
TriSquare(a′, b′, c′) = TriSquare(a, b, c) .
MRs of Insert-Sort, Bubble-Sort and Max–Min are same, described in MR7 and MR8:
MR7: If any two elements are exchanged with different values in the array, the programmes have the same result.
MR8: If all elements are reversed in the array, the programmes have the same results.

The Shortest-Path programme has two MRs (MR9 and MR10):

MR9: If in an undirected weighed graph, (S, v_1, v_2… v_n, D) is the shortest path, then S, v_1, v_2, …, v_n is the shortest path from node S to node v_n.
MR10: In an undirected weighed graph, the distance from node S to node D is equal to the distance from node D to node S.

The sin(x) programme has seven MRs (MR11-MR17). x represents the OTC and x′ represents the FTC.

MR11: If x′ = x + 2π, then sin(x′) = sin(x).
MR12: If x′ = x + π, then sin(x′) = − sin(x).
MR13: If x′ = − x, then sin(x′) = − sin(x).
MR14: If x′ = π − x, then sin(x′) = sin(x).
MR15: If x′ = 2π − x, then sin(x′) = − sin(x).
MR16: If x′ = 3 × x, then sin(x′) = 3 × sin(x) − 4 sin3(x).
MR17: If x′ = 5 × x then sin(x′) = 16 × sin5(x) + 5 × sin(3 × x) − 10 × sin(x).

Table 2. Parameter settings for FOA

Population Size	100
Area Limit	20
Lifetime	15
Transfer Rate	15
LSC	2
GSC	2
Lower Bound	1
Upper Bound	1000

Table 3. Programme Details

Programme	Structures	Number of targets
Triangle Area computation	Seven decision statements	6
Insertion Sort	One decision, three loops	24
Bubble Sort	One decision, two loops	30
Maxmin	two decision, one loops	30
Sin(x)	Five decisions, one loop	30
Shortest -Path	six decisions, six loops	30

6. RESULTS AND DISCUSSION

The comparative results using two fitness functions, traditional fitness (BD+AL) and ICF function with FOA are shown in Table 4, Table 5 and Table 6 respectively. Here, success rate (SR) and average time consumption (TC) is evaluated. SR is calculated for each path. In 100 runs how many times a path is covered successfully is calculated in percentage. TC is the total time consumption to generate test cases successfully in 100 runs. Table 7 shows one comparison result of FO+MR+ICF, Lv's method (Lv et al., 2018), Ahmed's method (Ahmed and Hermadi, 2008) and Yao's method (Yao &Gong, 2014).

In this paper, 3 comparison work are done

Table 4. Using Simple FOA (Triangle area computing problem)

	Traditional Fitness Function		ICF function	
Range	TC	SR%	TC	SR%
[1, 10]	0.720	96.45	0.110	100
[1, 50]	1. 194	96.23	0.544	100
[1, 100]	3.514	96.34	1.215	99.67
[1, 500]	6.349	89.78	3.972	98.54
[1, 1000]	11.834	89.56	6.967	96.65

Table 5. Using Simple FO +MR (triangle area computing Problem)

	Traditional Fitness Function		ICF function	
Range	**TC**	**SR%**	**TC**	**SR%**
[1, 10]	0.420	100	0.019	100
[1, 50]	0.882	100	0.104	100
[1, 100]	1.217	100	0.305	100
[1, 500]	2.694	100	0.363	100
[1, 1000]	5.733	100	0.409	100

(i) Comparison between traditional fitness function and ICF function in the context of test case generation and it is found in every case ICF gives better result in all respect (Shown in table 4, 5, 6).
(ii) Comparison between FOA based test case generation(Single path method) and FO-MRs based test case generation(Multipath method)

FO based test case generation shown in table 4, 5, 6 and 7.

Table 4 shows that Simple FOA does not give always 100% SR and TC is more than FO+MR because it takes more iterations than FO with MR. FO-MR gives 100% SR. Traditional fitness function take 11.834 sec and ICF take 6.967 sec where as in FO-MRs based in traditional it is 5.733 and in ICF 0.919 in triangle problem shown in Table 4.

Traditional fitness function in FO-MR takes less time consumption than ICF in simply FO based test case generation.

(iii) Comparison between FO-MR, PSO-MR and two GA based multipath coverage methods (Ahmed and Hermadi (2008); Yao and Gong et al. (2014)) are shown in table 7. FO-MR outperforms all the mthods. PSO-MR does not give 100% SR using traditional fitness function in triangle area computing problem and also in insertion sort, but in FO-MR gives 100% SR always. Yao's method performs better than Ahmed's method in every subject programme. Insertion sort, bubble sort and shortest path programme shows noticeable difference in time consumption.

Table 6. FO+MR all program

	Traditional Fitness Function		ICF function	
Programmes	**TC**	**SR%**	**TC**	**SR%**
Insertion Sort	25.195	97.83	0.789	100
Bubble Sort	2.682	100	0.332	100
Maxmin	1.322	100	0.684	100
Sin(x)	1.224	100	0.171	100
Shortest -Path	8.785	100	2.837	100

Table 7. Comparison results

Programmes	FO+MR+ICF	Lv's method(Lv et al., 2018)	Ahmed's method (Ahmed and Hermadi, 2008)	Yao's method (Yao and Gong, 2014)
	TC	TC	TC	TC
Triangle Programme	0.363	0.499	0.546	0.515
Insertion Sort	0.789	2.215	2.786	2.465
Bubble Sort	0.332	0.562	1.560	1.398
Maxmin	0.684	0.671	0.733	0.694
Sin(x)	0.171	0.234	0.343	0.295
Shortest -Path	2.837	3.258	5.287	4.478

To see the impact and efficiency of the MRs in test case generation, MRs are taken from different research papers (Lv et al., 2018; Chen et al., 2014, Kanewala et al., 2013; Dong et al., 2007). Individual MRs are chosen for test case generation as well as combinations of MRs of similar group and different group are chosen for test case generation. It is observed that

- It is observed that different MRs have different level of efficiency to cover the target path. For instance, in Table 8, relation 1(MR1) takes 0.655 sec to cover a single target path whereas relation 2 (MR4) covers two target paths in less time (0.631) than relation 1.
- The combination of MRs gives better result than the individual MRs . In Table 8 individually MR1 cover a single target path and MR4 covers two number of target paths. But the combination of MR1 and MR2 covers two target paths and also it takes less time for generation. More number of combinations give more efficiency.
- (MR1, MR2, MR3) and (MR4, MR5, MR6) of triangle programme having similar characteristics give many repeated computation (See Table 8) and combination of MRs of different group gives efficient result.
- FOA based MR traverse more paths than PSO based MR. For instance relation 13, 14, 15 covers 5 paths where in PSO based MR (Lv et al.,2018) it covers 4 paths.

6.1 Comparison Study with Existing Work

Every algorithm has some merits and demerits. Advantages of FOA over ABC, CS and Incremental GA is discussed here.

6.2 Comparison with ABC

Some limitations given below observed in ABC are overcome by FOA.

1. In ABC, one new solution can be generated from a current solution by changing the value of one random dimension which leads to slow convergence (Yazdani et al., 2014). In FOA, more than one dimensions are changed using LSC. Hence, it enhance the convergence speed. In ABC, random position is chosen without considering the fitness value which lowers the convergence speed wasting the fitness evaluations. In FOA, it considers the tree with best fitness value for adding its neighbor in local seeding stage.

Table 8. Performance of MRs in Triangle Area computing problem

Sl. No	MRs	TC	No. of paths covered
1	MR1	0.655	1
2	MR4	0.631	2
3	MR1 & MR2	0.620	2
4	MR4 & MR5	0.578	3
5	MR1 & MR4	0.6	3
6	MR1 & MR2 &MR3	0.630	2
7	MR4 & MR5 &MR6	0.535	4
8	MR1 & MR4 &MR5	0.535	4
9	MR1 & MR2 &MR4	0.533	4
10	MR1 & MR2 &MR4 &MR5	0.544	4
11	MR1 & MR4 &MR5 &MR6	0.544	4
12	MR1 & MR2 &MR3 &MR4	0.544	4
13	MR1 & MR2 &MR4 &MR5 &MR6	0.545	5
14	MR1 & MR2 &MR3 &MR4 &MR5	0.545	5
15	MR1 & MR2 &MR3 &MR4 &MR5 &MR6	0.556	5

2. In ABC, searching space is not adaptive to the current position of the whole swarm (Lakshminarayana et al., 2020; Kulkarni et al, 2011). In FOA, whole search space is considered by exchanging the values using global seeding operator.
3. In ABC, there is a possibility of abandonment of the best found position since this position has no possibility of improvement (Yazdani et al., 2014). The algorithm may lose the best solution. In FOA, It provides the possibility for the best tree to locally optimize its location by local seeding operator. Hence, there is no probability of abandonment of best found solution.

6.3 Comparison with CS

Though CS is a robust algorithm with fewer parameters, it creates one solution in one iteration by changing a single dimension only (Lin et al., 2016), which is overcome by LSC parameter of FOA. Hence, the coverage of FOA is faster than CS.

6.4 Comparison with IGA

Manikumar et al. (2016) propose Incremental GA (IGA) to generate test cases for branch coverage by adding an additional phase in GA. Our objective is to cover multiple paths in a single run and for that we have used metamorphic relations with Forest Optimization Algorithm (FOA).

Similarity between these two approaches are, they consist of two phases and the second phase takes input from the first phase to decrease the number of iterations. IGA decreases the number of iteration by decreasing the population. IGA considers the set of best chromosomes saved in each iteration of classical GA as its initial population. It targets to cover a single branch in each run and does the branch ordering following breadth first search. In our proposed approach, initial test case is generated using FOA, the successive test cases are generated using metamorphic relations without undergoing several runs. FOA is used as it generates candidate solutions exponentially in each iteration thus converges faster yielding best solution. Metamorphic testing generates test cases based upon the metamorphic relations avoiding the random generation of test cases which helps in covering multiple target paths faster.

7. THREATS TO VALIDITY

Threats to validity are the probable threats may present in the research work as all aspects of experiments may not be considered (Márcio Barros and Arilo Neto, 2011). Here, threats to research work is described in terms of external and internal threats.

7.1 Threats to Internal Validity

The following factors are the internal threats to our work.

- Parameter settings of searching algorithms may arise internal threats as it plays an important role in terms of time consumption and number of iterations required to cover a target path. Values of FOA parameters set based on other research in the proposed work. We have not experimented with some other values of the parameter that may enhance the performance.
- The second threat to validity is the selection and combination of MRs. It has a great impact on number of target path coverage. We select these MRs based on other research paper to compare our experimental results with other research work. Other combinations may give better results than the combinations used in this paper.

7.2 Threats to External Validity

The following factors are the threats due to external factors to our experimental results.

- Experiment with some large and complex programmes may show the extent of generalization of the proposed method. However, external threats are tried to decrease by taking six bench mark programs.
- Another external threat to validity is the selection of searching techniques. However, by observing the effective and efficient results of FOA in other areas, we select to apply the algorithm in test case generation.
- Selection of fitness function is another threats to external validity. However, we tested the consistency of ICF fitness function by solving a linear complex problem (SSSP) in our previous research work (Sahoo & Ray, 2020) and six programs in this paper.

8. CONCLUSION AND FUTURE WORK

In this paper, code based test case generation for multiple paths is considered. Most of research work have been done on test case generation for single path coverage. It suffers from more number of iterations and more execution time and less number of target path coverage. To overcome the problems of single path coverage, we propose MR based test case generation using FOA. The motive of using this algorithm is that this algorithm itself covers more number of paths in a single run as shown in table 5. The combination of FOA and MRs give more efficient result in terms of time consumption and number of path coverage. The OTC is generated by FOA and FTCs are generated by iterative execution of MRs. If the MRs stuck in some target path, FOA gives the lift and covers more number of target paths efficiently. Experimental results prove that the combination of MRs, FOA and ICF gives better results than other techniques used in this paper.

Future work is to explore the proposed approach with other code coverage criteria like MC/DC coverage criteria. MRs are mainly designed for fault detection in oracle problem. As far best of our knowledge combination of meta heuristic techniques and metamorphic relations have not been used for fault detection. Future work will be fault detection using the combination of these two techniques.

REFERENCES

Ahmed, M. A., & Hermadi, I. (2008). GA-based multiple paths test data generator. *Computers & Operations Research*, *35*(10), 3107–3124. doi:10.1016/j.cor.2007.01.012

Arcuri, A., & Fraser, G. (2014, August). On the effectiveness of whole test suite generation. In *International Symposium on Search Based Software Engineering* (pp. 1-15). Springer. doi:10.1007/978-3-319-09940-8_1

Asrafi, M., Liu, H., & Kuo, F. C. (2011, June). On testing effectiveness of metamorphic relations: A case study. In *2011 Fifth International Conference on Secure Software Integration and Reliability Improvement* (pp. 147-156). IEEE. doi:10.1109/SSIRI.2011.21

Baresel, A., Sthamer, H., & Schmidt, M. (2002, July). Fitness function design to improve evolutionary structural testing. In *Proceedings of the 4th Annual Conference on Genetic and Evolutionary Computation* (pp. 1329-1336). Academic Press.

Chaghari, A., Feizi-Derakhshi, M. R., & Balafar, M. A. (2018). Fuzzy clustering based on Forest optimization algorithm. *Journal of King Saud University-Computer and Information Sciences, 30*(1), 25-32.

Chen, T. Y., Cheung, S. C., & Yiu, S. M. (1998). *Metamorphic testing: a new approach for generating next test cases.* arXiv preprint arXiv:2002.12543.

Chen, Y., Zhong, Y., Shi, T., & Liu, J. (2009, August). Comparison of two fitness functions for GA-based path-oriented test data generation. In *2009 Fifth International Conference on Natural Computation* (Vol. 4, pp. 177-181). IEEE. doi:10.1109/ICNC.2009.235

de Oliveira Barros, M., & Dias-Neto, A. C. (2011). 0006/2011-threats to validity in search-based software engineering empirical studies. *RelaTe-DIA, 5*(1).

Ghaemi, M., & Feizi-Derakhshi, M. R. (2016). Feature selection using forest optimization algorithm. *Pattern Recognition*, *60*, 121–129. doi:10.1016/j.patcog.2016.05.012

Ghaemi, M., & Feizi-Derakhshi, M. R. (2014). Forest optimization algorithm. *Expert Systems with Applications*, *41*(15), 6676–6687. doi:10.1016/j.eswa.2014.05.009

Gong, D. W., & Zhang, Y. (2010). Novel evolutionary generation approach to test data for multiple paths coverage. *Dianzi Xuebao*, *38*(6), 1299–1304.

Harman, M., Jia, Y., & Zhang, Y. (2015, April). Achievements, open problems and challenges for search based software testing. In *2015 IEEE 8th International Conference on Software Testing, Verification and Validation (ICST)* (pp. 1-12). IEEE. doi:10.1109/ICST.2015.7102580

Kanewala, U., & Bieman, J. M. (2013, November). Using machine learning techniques to detect metamorphic relations for programs without test oracles. In *2013 IEEE 24th International Symposium on Software Reliability Engineering (ISSRE)* (pp. 1-10). IEEE. doi:10.1109/ISSRE.2013.6698899

Khanna, M., Chauhan, N., & Sharma, D. K. (2019). Search for Prioritized Test Cases during Web Application Testing. *International Journal of Applied Metaheuristic Computing*, *10*(2), 1–26. doi:10.4018/IJAMC.2019040101

Khanna, M., Chauhan, N., Sharma, D., Toofani, A., & Chaudhary, A. (2018). Search forioritized test cases in multi-objective environment during web application testing. *Arabian Journal for Science and Engineering*, *43*(8), 4179–4201. doi:10.1007/s13369-017-2830-6

Kulkarni, N. J., Naveen, K. V., Singh, P., & Srivastava, P. R. (2011, July). Test case optimization using artificial bee colony algorithm. In *International conference on advances in computing and communications* (pp. 570-579). Springer.

Lakshminarayana, P., & SureshKumar, T. V. (2020). Automatic Generation and Optimization of Test case using Hybrid Cuckoo Search and Bee Colony Algorithm. *Journal of Intelligent Systems*, *30*(1), 59–72. doi:10.1515/jisys-2019-0051

Lin, Y., Zhang, C., & Liang, Z. (2016). Cuckoo search algorithm with hybrid factor using dimensional distance. *Mathematical Problems in Engineering*, *2016*, 2016. doi:10.1155/2016/4839763

Liu, F., Huang, H., Yang, Z., Hao, Z., & Wang, J. (2019). Search-Based Algorithm With Scatter Search Strategy for Automated Test Case Generation of NLP Toolkit. *IEEE Transactions on Emerging Topics in Computational Intelligence*, 1-13.

Lv, X. W., Huang, S., Hui, Z. W., & Ji, H. J. (2018). Test cases generation for multiple paths based on PSO algorithm with metamorphic relations. *IET Software*, *12*(4), 306–317. doi:10.1049/iet-sen.2017.0260

Mahmoudi, S., & Lailypour, C. (2015). A discrete binary version of the Forest Optimization Algorithm. *International Conference on Information Technology, Computer & Communication*.

Mall, R. (2018). *Fundamentals of software engineering*. PHI Learning Pvt. Ltd.

Manikumar, T., Kumar, A. J. S., & Maruthamuthu, R. (2016). Automated test data generation for branch testing using incremental genetic algorithm. *Sadhana*, *41*(9), 959–976. doi:10.1007/s12046-016-0536-1

Mayer, J., & Guderlei, R. (2006, September). An empirical study on the selection of good metamorphic relations. In *30th Annual International Computer Software and Applications Conference (COMPSAC'06)* (Vol. 1, pp. 475-484). IEEE. doi:10.1109/COMPSAC.2006.24

Mohanty, S., Patra, P. K., Ray, M., & Mohapatra, S. (2018). A novel meta-heuristic approach for load balancing in cloud computing. *International Journal of Knowledge-Based Organizations*, *8*(1), 29–49. doi:10.4018/IJKBO.2018010103

Panichella, A., Kifetew, F. M., & Tonella, P. (2017). Automated test case generation as a many-objective optimisation problem with dynamic selection of the targets. *IEEE Transactions on Software Engineering*, *44*(2), 122–158. doi:10.1109/TSE.2017.2663435

Sahargahi, V., & Derakhshi, M. F. (2017). Course Timetabling Using Forest Algorithm. *International Journal of Computer Science and Network Security*, *17*(2), 83–93.

Sahoo, R. R., & Ray, M. (2020). PSO based test case generation for critical path using improved combined fitness function. *Journal of King Saud University-Computer and Information Sciences*, *32*(4), 479–490. doi:10.1016/j.jksuci.2019.09.010

Sahoo, R. R., & Ray, M. (2018). Metaheuristic techniques for test case generation: A review. *Journal of Information Technology Research*, *11*(1), 158–171. doi:10.4018/JITR.2018010110

Suji, R. J., & Rajag, S. P. (2016). Multi-ranked feature selection algorithm for effective breast cancer detection. *Biomedical Research*, 99-102.

Xie, Q., Cheng, G., Zhang, X., & Peng, L. (2020). Feature Selection Using Improved Forest Optimization Algorithm. *Information Technology and Control*, *49*(2), 289–301. doi:10.5755/j01.itc.49.2.24858

Yazdani, D., & Meybodi, M. R. (2014, October). A novel artificial bee colony algorithm for global optimization. In *2014 4th International Conference on Computer and Knowledge Engineering (ICCKE)* (pp. 443-448). IEEE. doi:10.1109/ICCKE.2014.6993393

Yao, X., & Gong, D. (2014). Genetic algorithm-based test data generation for multiple paths via individual sharing. *Computational Intelligence and Neuroscience*, *2014*, 2014. doi:10.1155/2014/591294 PMID:25691894

Zhang, J., Chen, J., Hao, D., Xiong, Y., Xie, B., Zhang, L., & Mei, H. (2014, September). Search-based inference of polynomial metamorphic relations. In *Proceedings of the 29th ACM/IEEE international conference on Automated software engineering* (pp. 701-712). doi:10.1145/2642937.2642994

Zhu, Z., Xu, X., & Jiao, L. (2017, June). Improved evolutionary generation of test data for multiple paths in search-based software testing. In *2017 IEEE Congress on Evolutionary Computation (CEC)* (pp. 612-620). IEEE.

Rashmi Rekha Sahoo received her M.Tech. in Information Technology from Siksha 'O' Anusandhan Deemed to be University in 2012. Now, she is pursuing her Ph.D in Siksha 'O' Anusandhan Deemed to be University. Her research area is Software Testing.

Mitrabinda Ray received her M.Tech. in Computer Science in the year 2003 and Ph.D degree from NIT, Rourkela in 2012. Her research area is Software Testing, Reliability Analysis and Risk Estimation. Currently, she is working as Associate Professor in Computer Science & Engineering Department in Siksha 'O' Anusandhan Deemed to be University.