*I.J. Mathematical Sciences and Computing,* **2015, 3, 1-8**
Published Online September 2015 in MECS (http://www.mecs-press.net)
DOI: 10.5815/ijmsc.2015.03.01

***Available online at http://www.mecs-press.net/ijmsc***

# The Application of Meta-Heuristic Algorithms in Automatic Software Test Case Generation

Maryam Mirzapour Moshizi[a], Amid Khatibi Bardsiri[b]

[ab] *Department of Computer Science, Bardsir Branch, Islamic Azad University, Kerman, Iran*

**Abstract**

Nowadays, software test is one of the most important activities that software's quality will be certified by it. Test operation includes program's implement on test case set and comparison of results with expected one. Manual test case for operation test program and error detect is time consuming with insufficient precision and complicated coverage of program, so, the use of algorithms in automatic test case generation has been considered. Meta-heuristic algorithms are known tools which are optimized and used in test case generation. Most of complicated matters need a lot of possible states assessment in order to reach the valid answer. With the proper answer, test case optimization and meta-heuristic algorithms play a constructive role. In this paper we would compare methods and their traits, and the software test case generation methods based on meta-heuristic algorithms with their description.

**Index Terms:** Test Case Generation, Meta-Heuristic Algorithms, Software Test Case

## 1. Introduction

Nowadays, software is being produced rapidly, and its quality is one of the important factors in software industry so, they must be tested in order to achieve a good quality. Software test is one of the most crucial and acute must to be activities during the software project. Validation and accuracy will affect on result of software test quality directly, so schematization and accurate implement of software test would have a defining role for failure or achievement. Software test will use 30% of project endeavour in most of tasks. In acute safety application, software test can use 50% to 80% of project endeavour [1]. Defective software is usually costly and software test is the only way for defect finding. Of course, we should consider that test can show errors not inexistence. Because not to be any error doesn't mean there is no error in the program. There have been made a lot of techniques for software test that their purpose is exhibition of a successful method which at least could be

* Corresponding author. Tel.:
E-mail address: mmairr_2007@yahoo.com； a.khatibi@srbiau.ac.ir

able to find undetected error. Nowadays it's common to generate automatic test case for software experiments. Software based search with use of meta-heuristic search technique has been already considered that does test automatic or semiautomatic by many researchers [2]. By the way, it's better to know that no test is complete and can't insure lack of error just one method. The new methods for software test try to do test in a proper time with perfect coverage [3]. Then we introduce criterions for the survey of some automatic software output and test method with meta-heuristic algorithms.

## 2. Automated Test Case Generation Algorithms Comparison Criteria

In this paper 8 methods that used meta-heuristic algorithms in order to generation automatic test case have been studied. Their characteristic studied by the use of these criteria.

Algorithms assessment criteria:

- Meta-heuristic algorithm: These criteria specify that meta-heuristic algorithm has been used for strategy advancement.
- Test case generation/ Test case priority: The purpose of algorithm can be specified by these two methods. Nowadays, beside the test case generation, priority is considered, too.
- Start of algorithm: Lots of test case generation methods start with a random set of test case and optimization accomplishes on this set.
- Benefits of algorithm's use: Except test case generation, each algorithm has some benefits that the most important one is perfect coverage of program. The other benefits include: reduction of frequency number, reduction of performance time, instant access to solution, improve of time and place complexity.

## 3. Automated Test Case Generation Algorithms

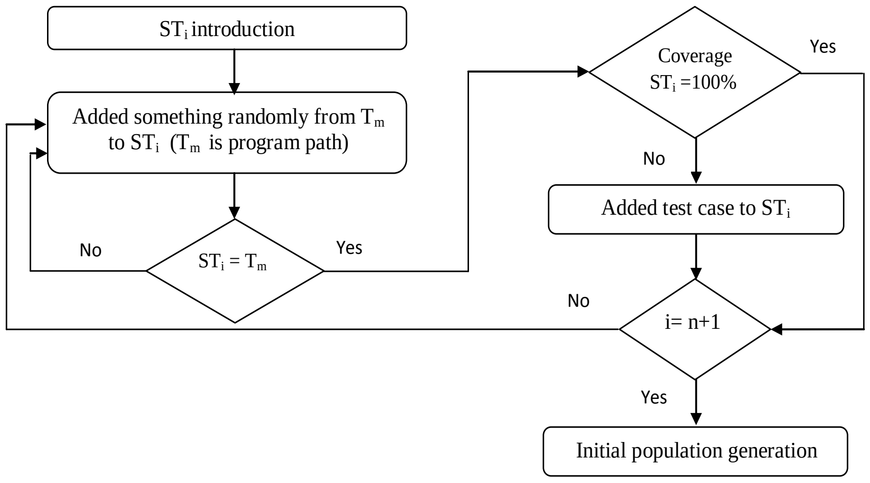
$ST_i$ introduction
Added something randomly from $T_m$ to $ST_i$ ($T_m$ is program path)
$ST_i = T_m$
No
Yes
Coverage $ST_i$ =100%
Yes
No
Added test case to $ST_i$
No
i= n+1
Yes
Initial population generation

Fig. 1. Initial population generation stages

### 3.1 TCP_GA[1]

[1] Test Case Prioritization Technique based on Genetic Algorithm

In this method, for the effect enhancement of test, test calibration techniques permit the tester to prioritize test items. Genetic Algorithm is used for this method, according to fig1 the initial population generates.

After former the initial population, the amount of fitness function test case would be estimated and it is between 0 and 1. This amount shows their adaptability is in solution area. The higher amount of Fitness function the most probability of that test frequency. After that the selection method is done and uses Roulette Wheel strategy and the proportion of fitness value to total is accomplished.

$$Ratio(i) = \frac{f(i)}{(f(1) + f(2) + ... + f(s))} \tag{1}$$

S is the number of individual in population. After arranging the population individual based on fitness value, random number r Є [0,1] produces and then the individuals that their fitness set is more or equal to r will be selected. This process continues up to the number of population equal to s. the next generation will be produced and the more fitness function, the more opportunity for being in next generation. After selection, crossover and mutation operation will be accomplished and produce the new population. When iteration is more than a special value, optimal answer returns. This algorithm has been compared with common genetic algorithm and the results shoe this algorithm is faster for reaching to answer [4].

## 3.2 GA_PSO[2]

In this paper, the hybrid algorithm has been used. Genetic algorithm and particle swarm optimization are demonstrated. In this algorithm, a preliminary random population (p1), that algorithm will be applied on it and one population with vacant values (p2), that PSO algorithm needs it for assessment, will be produced. For P1 fitness function value will be evaluated for each chromosome and is based on the farness between particles path and purpose path. In case the actual path covers the purpose path completely or frequency numbers reach to thresh hold, algorithm will terminate and optimal test case would be produced. Otherwise, p1 members will be arranged based on fitness function value and choose the best particle of p1 and added to p2. Then PSO algorithm will be applied for p2 and we’ll have a new population of p2in this phase. For each member of new p2 population we upgrade fitness function value and if the optimal response isn’t being achieved, we will rebuild it. In this part the genetic algorithm will be used in order to optimize p1 population. According to crossover and mutation, the new p1 population will be achieved. All mentioned ways are repeated. With this survey, it’s formed that the average of frequency and time performance decrease. This algorithm has been compared with GA and PSO and it’s turned out that it will find the optimal answer faster [5].

## 3.3 BCO[3]

In this algorithm, bee colony strategy has been used. All bees aren’t search for the food on the contrary random number is searching for food. ABC[4] algorithm can be used as parallel, so some factors are explained for this purpose and form it.

Algorithm stages:

1- Test cases are produced randomly (food origin).
2- All of test cases are evaluated for the program.

2 AUTOMATIC Path Test Data Generation Based on GA-PSO
3 Automated Generation of Independent Path and Test Suite Optimization Using Artificial Bee Colony
4 Artificial Bee Colony

3- If the criteria aren’t satisfying, they would be done next stage, otherwise all test cases would be produced and algorithm would stop.
4- All independent ways are grouped in SQRT list
5- One thread is attended in each set and Bee Thread algorithm is applied for all threads.
6- If one clear thread was convincing for all independent ways, thread would be determined. But if none of threads is able to solve one new independent way, all ways would be compounded with each other and apply for a repetition cycle, so the independent ways between parallel threads would spread. If thread found one new way, it would continue the process.
7- In this stage, it’s checked that the new independent way would cover all criteria or not. Otherwise, the Bee Tread algorithm would be applied and return to stage 6.

In this method, all independent bees move as parallel in order to reach to purpose. Parallel and independent behavior of BCO, make test case generation efficiently and rapidly. This method reduces problems of evaluation and memory’s limitation. In comparison to ACO and GA, ABC improves time and memory troubles and covers the program with the least numbers and frequency of test [6].

### 3.4 GA-A[5]

In this method some changes will happen to GA in order to reach to the optimal answer. Unlike the other algorithms, GA searches all spaces and chooses the samples randomly, evaluates criteria, applies crossover and mutation in order to optimize the samples and finally produces the optimal answer. Overall, GA population has been chosen randomly, but for population introduction, it’s better to name a different beneficial method. After population choice, information must become encode and change to string bits. String bit divides into L. the first parts have a higher position and the last parts have a lower position. This encode method is more appropriate. In genetic algorithm, selection phase is after population introduction. Selection is one probability of crossover and mutation (pci, pmi)for each individual as the worst individuals don’t have opportunity for going to the next level.

After this phase, we have assessment. Assessment function must be able to assess the produced generation well and find purpose parameters based on GA, Branch Function is used for this aim and covers all spaces and path. The result of this algorithm shows the high speed of test case generation and the most perfect coverage on program. This method has a better effect than ordinary GA [7].

### 3.5 HST[6]

In this method, GA and search algorithm, Tabu, is used. Tabu search divides the test cases into two groups: Long term and Short term.

Long term test cases have more effects for error detection. Test case generation is randomly in ordinary GA. Sometimes test cases’ continuation has been produced unintentionally. In order to prevent from this, hybrid algorithm id=s used. Firstly GA is used by this method and its phases continue up to the end of crossover but mutation phase uses Tabu algorithm. The most frequently used test cases are in Short term list. These two tables are used for the continuation of new test cases. These test cases find the errors efficiently. In the other words, they find the errors during the shortest possible time. The coverage needed tie is estimated for each test case and phase. The best test case is produced by GA and the new ones are produced by Tabu search algorithm and optimal test cases are produced by them. This algorithm is more efficient and constructive in comparison to the pertinent algorithms [8].

---

5 Genetic Algorithm and its Application in Software Test Data Generation
6 Test Case Optimization Using Hybrid Search Technique

## 3.6 OTF[7]

Firefly algorithm is one of the optimal mentioned algorithms. The radical idea is generated from luminescent relation of fireflies. The firefly pays attention to luminosity of the other fireflies. This algorithm is used to find test cases. Objective function is used for this matter and decreases the search algorithm, determines 1- cyclomatic complexity 2- random function.

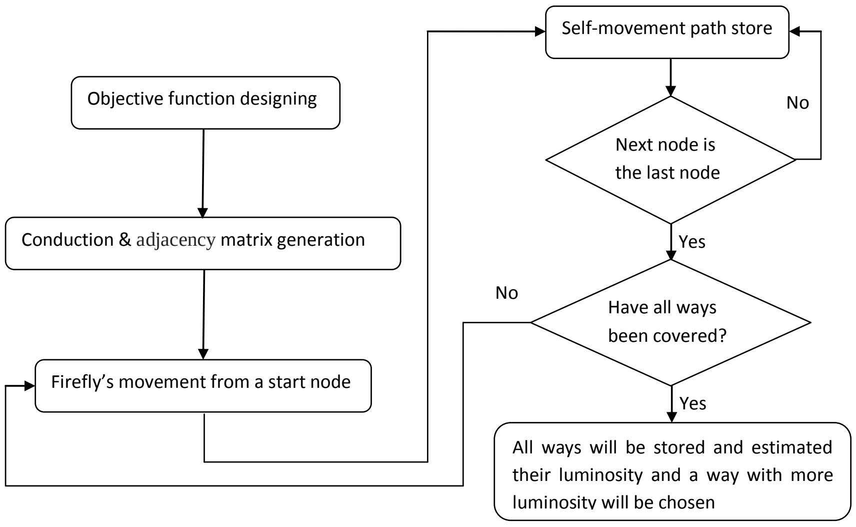
Objective function designing
Conduction & adjacency matrix generation
Firefly's movement from a start node
Self-movement path store
Next node is the last node
No
Yes
Have all ways been covered?
No
Yes
All ways will be stored and estimated their luminosity and a way with more luminosity will be chosen

Fig. 2. OTF algorithm stages

Adjacency matrix and conduction matrix lead the fireflies toward optimal answers. The light intensity will decrease will decrease on moving way. Finally a way will be chosen that with complete software coverage finds brilliance. In comparison to ACO algorithm, this algorithm has a better coverage on the program. It optimizes test cases and decreases their numbers and attempt [9].

## 3.7 TSBCO[8]

In this method bee colony algorithm has been used. Firstly solutions have randomly produced. Then the ways will be found based on neighbor and accessible mechanisms. In this method test data is produced based on white box techniques and chooses the test greedy. In this phase, test cases have fitness and will be chosen with the higher amount of fitness. This process will repeat in order to produce test cases with 100% fitness. Finally the generated test cases will be stored in database.

This algorithm produces optimal test cases as well as it does test cases priority. Coverage of this method is well and produces the solutions faster [10].

7 Test Case Optimization Using Hybrid Search Technique
8 Test Case Selection Using Bee Colony Optimization

## 3.8 TCP-GA-TP[9]

In this method, GA has been used. We consider a remarkable number betweens 20 to 150 and a fitness function betweens 0 to 1 for each chromosome. The chromosome with higher fitness function maybe seen in next generation. The black box test techniques are used here and divide test into some parts which are called test point. This algorithm designs test cases according to test point and finally we'll have test cases set. For each test case set, APTC[10] assesses the percentage of test point coverage. APTC's concentration is on test point coverage. The more test point, the more error's finding. Finally priority and order of test cases can be determined by test case's use [11].

This method has a complete coverage on program.

Table 1. Automated test case generation algorithm comparison

| ***Algorithm's name*** | ***Meta-heuristic algorithm*** | ***Test case generation*** | ***Test case priority*** | ***Algorithm start*** | ***Advantage of algorithm*** |
|---|---|---|---|---|---|
| ***TCP-GA*** | *Genetic algorithm* | √ | √ | *Random test case set* | *Faster generation and more perfect coverage in comparison to GA* |
| ***GA-PSO*** | *Genetic algorithm and Particle Swarm Optimization algorithm* | √ | | *Random test case set* | *Reduction the average of frequency number and performance time in comparison to GA and PSO* |
| ***BCO*** | *Bee Colony Optimization algorithm* | √ | | *Random test case set* | *Improvement in time and memory complexity in comparison to ACO and GA and more perfect coverage and reduction the average of frequency number* |
| ***GA-A*** | *Genetic algorithm* | √ | | *Random test case set* | *Faster generation and more perfect coverage in comparison to GA* |
| ***HST*** | *Genetic algorithm and Tabu search algorithm* | √ | | *Random test case set* | *More perfect coverage in comparison to GA* |
| ***OTF*** | *Firefly algorithm* | √ | | *Objective Function* | *more perfect coverage in comparison to ACO and reduction of test cases and test effort* |
| ***TSBCO*** | *Bee Colony Optimization algorithm* | √ | √ | *Random test case set* | *Faster generation and more perfect coverage* |
| ***TCP-GA-TP*** | *Genetic algorithm* | √ | √ | *Random test case set* | *More perfect coverage* |

## 4. Table's Analysis

With due attention to this table, it is found that GA, Bee Colony, Particle Swarm Optimization, Firefly in 8 methods have been used. All strategies purpose is test case generation and it's seen that three of them beside of this purpose, prioritize test case too. The algorithm start is in the 5th column and in most of them it's started with a random initial set and in algorithm phases, optimal test case set are generated. In the last column, the advantages of each strategy are seen. Complete coverage is the most important one. Reduction of program's frequency, reduction of time implement, faster achievement to solution and improvement of time and memory complexity are its advantages. These advantages are achieved by the comparison methods with other

9 Test Case Prioritization Based on Genetic Algorithm and Test-Points Coverage
10 Average Percentage of Test Point Coverage

algorithms and aren't absolute.

## 5. Conclusions

Nowadays, with fast improvement of software, their quality's guarantee is one of the most important activities in software life time. There are many ways for the test of software quality that includes program implement on set of test cases and result comparison. The purpose of test is to find the errors of program. The manual test cases are time-consuming with insufficient accuracy and moreover program coverage is hard by them. So, automated test case generation algorithms are produced. Although many methods have been suggested, meta-heuristic algorithm is used too. In this paper, 8 methods of automated generation of test case with the application of meta-heuristic algorithm have been studied and 5 criteria are mentioned for them.

According to Table1, it's found that majority of algorithms have good coverage on software and one of the most important purposes is the coverage because it's easier to find errors by it. Error finding is important. Remember that no test is complete.

## References

[1] Collofello, J. , Vehathiri, K, "An environment for training computer science students on software testing", Frontiers in Education, 2005. FIE '05. Proceedings 35th Annual Conference, pp T3E 6-T3E 10

[2] McMinn, " Search-Based Software Testing: Past, Present and Future", Software Testing, Verification and Validation Workshops (ICSTW), Fourth International Conference, 2011, pp 153-163

[3] Phil McMinn, "Search-based software test data generation" Software Testing, Verification and Reliability, Volume 14, Issue 2, 2004 , PP 105–156

[4] Wang Jun, Zhuang Yan, Jianyun Chen, "Test Case Prioritization Technique based on Genetic Algorithm", International Conference on Internet Computing and Information Services, 2011, pp 173-175

[5] Sheng Zhang, Ying Zhang, Hong Zhou, Qingquan He, "Automatic Path Test Data Generation Based on GA-PSO" Intelligent Computing and Intelligent Systems (ICIS) International Conference, 2010, pp 142-146

[6] Soma Sekhara Babu lam, "Automated Generation of Independent Path and Test Suite Optimization Using Artificial Bee Colony", Procedia Engineering, Volume 30, 2012, pp 191–200

[7] Wang Lijuan, Zhai Yue, Hou Hongfeng, "Genetic Algorithms and Its Application in Software Test Data Generation", International Conference on Computer Science and Electronics Engineering, 2012, pp 617-620

[8] J.Albert Mayan, T Ravi, "Test Case Optimization Using Hybrid Search Technique"**,** International Conference on Interdisciplinary Advances in Applied Computing, 2014, pp 1-7

[9] Praveen Ranjan Srivatsava, B. Mallikarjun, "Optimal test sequence generation using firefly algorithm", Swarm and Evolutionary Computation, Volume 8, 2013, pp 44-53

[10] Navdeep Koundal, Ankur Shukla, Mohsin Rashid Mir, "Test Case Selection Using Bee Colony Optimization", International Journal of Science and Research (IJSR), Volume 3, Issue 5, 2014, pp 1432-1436

[11] Weixiang Zhang, Bo Wei, Huisen Du, " Test Case Prioritization Based on Genetic Algorithm and Test-Points Coverage", Algorithms and Architectures for Parallel Processing, volume 8630, 2014, pp 644-654.

## Authors' profiles

**Maryam Mirzapour Moshizi** received a B.S. degree in computer science from Shahid Bahonar university in 2009, and M.S. degree in computer software engineering from IAU university in 2015. She works on Software testing and intelligent algorithms.

Email: mmairr_2007@yahoo.com

Address: Bardsir Branch, Islamic Azad University, Kerman, IRAN

**Amid Khatibi Bardsiri** ***(corresponding author)*** received his B.S. degree in computer software engineering from Shahid Bahonar university, Kerman, Iran in 2008, and his M.S. degree in software engineering from Islamic Azad University, Tehran, Iran, in 2010. He is currently undertaking his PhD in science and research branch of Islamic Azad University, focusing on software metrics and measurement. He published about 25 research papers in international journals and conference proceedings. His areas of research include information systems engineering, software development, software metrics, grid computing, and cloud computing.

Email: a.khatibi@srbiau.ac.ir

Postal address: Fajr 12 Ave, 22 Bahman Blvd, Bardsir, Kerman, IRAN

Postal code: 7841634367

**How to cite this paper:** Maryam Mirzapour Moshizi, Amid Khatibi Bardsiri,"The Application of Meta-Heuristic Algorithms in Automatic Software Test Case Generation", International Journal of Mathematical Sciences and Computing(IJMSC), Vol.1, No.3, pp.1-8, 2015.DOI: 10.5815/ijmsc.2015.03.01