ISSN No. 0976-5697

Volume 4, No. 10, Sep-Oct 2013

International Journal of Advanced Research in Computer Science

RESEARCH PAPER

Available Online at www.ijarcs.info

Particle Swarm Optimization Algorithm for Randomized Unit Testing

K. Devika Rani Dhivya M.Sc (CS)., M.Phil.,
Assistant Professor, Dept of CA and SS,
Sri krishna Arts and Science College,
Coimbatore, India.
Devika58@ gmail.com

Abstract: Randomized testing is an effective method for testing units of software. Thoroughness of randomized unit testing is according to the settings of optimal parameters. For the purpose of checking the test input data the randomized testing uses randomization. Designing Genetic algorithm is somewhat of a black art. The feature subset selection (FSS) tool is used with genetic algorithm to reduce the size and the content of the test case. The existing system does not cover all test cases because it can quickly generate many test cases and does not consider the target method. Thus GA for Randomized Unit Testing has not achieves high test coverage and does not produce better optimal test data. In this paper, particle swarm optimization (PSO) algorithm is used for randomized unit testing. PSO algorithm is used to evaluate the target method solutions for test coverage in test data. The main goal is to generate the optimal test parameter, reduce the size of test case and to achieve high coverage of the testing units. PSO achieves high coverage and produces the better optimal value within 20% of the time with better accuracy.

Keywords: Randomized unit testing, Genetic algorithm, Feature Sub Set Selection, PSO algorithm.

I. INTRODUCTION

Software testing is the process of examining a software product to find errors. Testing involves running a piece of software for a set of input data and checking the outputs for correctness [7]. The unit testing is the process of testing each and every module individually [12].The goal of software testing is to evaluating the quality of an application and to improve it a primary purpose of testing is to detect software failures so that defects may be discovered and corrected. The action and the accuracy performance of randomized unit testing is depends up on applying randomization, for e.g. the number of method calls are made, the relative frequency with which different methods are called and the ranges from which numeric arguments are chosen [7]. Randomization for some aspect of test input is used by randomized testing process. Randomized testing of a unit is very effective at forcing failures in even a well-tested unit [4][5][6][7][13].

PSO is used for Randomized unit testing. It is an optimal search algorithm where test data sets are collectively called as swarms. The main function of PSO is to produce an optimal test data for randomized unit testing.

This paper describes the optimization algorithm PSO which is used for randomized unit testing process. The PSO algorithm achieves High test case coverage (90%) within less time.

A. Randomized Unit Testing:

Random testing can quickly generate many tests, is easy to implement, scales to large software applications, and reveals software errors. For selecting the test input data randomized testing use randomization. Random test data generators select random inputs from test data from some distribution. The main benefit of randomized unit testing is the ability to produce many test inputs in a less time, it also includes test inputs that may not be selected, but which may become force failures [7].

Randomized unit testing is depends up on the generating many input test data, which is difficult to check by a tester. Randomized unit testing is a well-known technology that has been shown to be very effective, but efficiency is depends on the settings of test optimization algorithm parameters. To solve the issues of the randomized unit testing, thus generate the test data and test case generator by use the following algorithm to more coverage of the test cases. The algorithms are GA, and PSO. These two algorithms are optimization search algorithm; methods are used to produce an optimization test data and coverage of the test case at highest level.

B. Test Case And Test Data Generation:

A test case is a set of condition or variables under which software tester will determine whether the application or a system is working correctly [7].

The value, which is given as input for the application is a test data. Every test data is assumed under some conditions for the correctness of the application, thus the test case is generated. The algorithm takes two parameters to construct the test data and test case generation: a set M of java target methods and a genetic algorithm chromosome c appropriate to M. The method call is used to constructed and run test cases under randomized unit testing in genetic algorithm [1][2]. A test case is defined as the sequence of call conditions and structural descriptions, the test case generation takes the possible test data as a solution to search, and applies it to optimization search algorithm to find for a good optimized test data [9]. The test case generation collects the number of lines covered by the test case.

C. Genetic algorithm:

Genetic algorithm first described by Holland [7]. Candidate solutions are presented as “chromosomes”, with solution represented as “genes” in the chromosomes. The possible chromosomes form a search space and are associated with a fitness function representing the value of solutions encoded in the chromosome. Search proceeds by evaluating the fitness of each of a population of chromosomes, and then performing point selection, mutations and recombination on the successful chromosomes.

The GA represents the design variables of each individual design with binary strings of 0's and 1's that are referred to as chromosomes [14]. The GA begins its search from a randomly generated population of designs that evolve over successive generations (iterations).

The first operator is the "Selection" operator that mimics the principal of "Survival of the Fittest". The second operator is the "Crossover" operator propagates features of good surviving designs from the current population into the future population, which will have better fitness value on average. The last operator is "Mutation", which promotes diversity in population characteristics. The mutation operator allows for global search of the design space and prevents the algorithm from getting trapped in local minima.

D. Genetic algorithm with feature subset selection:

The feature subset selection (FSS) tool is used with genetic algorithm. The purpose of this, Feature subset selection tool is used to assess and to reduce the size and content of the test case within the genetic algorithm [7][8]. The tool is used to achieve highest of the test coverage of test data in a test case [6]. It is a data mining technique it removes the unwanted or needless information. It automatically find and remove superfluous parts of the search control and it is also used to prune the needless information in the gene types and achieving high level of the nearly same coverage.

E. Particle Swarm Optimization (PSO):

The concept of PSO was first introduced by Kennedy and Eberhart in 1995. Particle swarm optimization (PSO) has been shown as an effective tool for solving global optimization problems **[**3**].** Particle swarm optimization (PSO) is inspired by the social behavior observed in flocks of birds and schools of fish. The particles are initially distributed randomly over the search space with a random velocity and position. Here, the goal is to fine the global optimum of the test case or a system [10]. In nature, there is a leader who leads the bird or fish group to move. In PSO, a potential solution to the considered problem is represented by a particle, similar to the individuals in the bird and fish group.PSO is a population-based stochastic optimization paradigm, in which each agent, named particle, of the population, named swarm, is thought of as a collision-proof bird and used to represent a potential solution. PSO is similar in some ways to other existing GAs, such as GA, but is defined in a social context as opposed to a biological context. Compared to GA PSO is generally characterized as simple in concept, easy to implement and computationally efficient for coverage of the tests cases and reduces the computational cost of the GA.

II. RELATED WORK

James Miller, Marek Reformat, Howard Zhang [2006], in this paper they proposed a new approach utilizing program dependence analysis techniques and genetic algorithms (GAs) to generate test data. Testing costs often account for up to 50% of the total expense of software development; hence any techniques leading to the automatic generation of test data will have great potential to considerably reduce costs. [2]

James H. Andrews, Susmita Haldar, Yong Lei and Felix Chun Hang Li [2006], in this paper they proposed a RUTE-J and they illustrate how it supports and development of pre-unit solution for the problem of randomized unit testing. The use of Randomized testing in experimentation by adapting RUTE-J is defined. Thus it performs effective method of forcing failure,. In this technique, the test coverage area will be minimized up to the test level of the suite, thus it improve in speed of testing. [1]

James H. Andrews and Felix C. H. Li [2007], they proposed the randomized testing which is more effective method for testing a software. They describe the process of Genetic algorithm which gives an optimal parameter. Here, they describe about the implementation of "A novel two-level genetic random testing system "Nighthawk. This paper results, that randomized unit testing is a effective technology, but whose efficiency depends up on the settings of the testing optimal algorithm test data, thus they describe the nighthawk, a system which an upper level genetic algorithm and lower level randomized unit testing concept. This tool, which achieves a good test coverage and produce optimal result. [4]

James H. Andrews and Tim Menzies [2009], in this paper they try to improve the speed and coverage of the system, a model generation to meet the size and time constraints. There will always be a trade-off between completeness and runtime speed, thus they explore that trade-off in the context of using genetic algorithms to learn coverage models, which are in the control structures for randomized test generators. After applying feature subset selection to logs of the GA output, we find we can generate the coverage model and run the resulting test suite ten times faster while only losing 6% of the test case coverage. They applied feature subset selection (FSS) to a metaheuristic algorithm used to learn coverage models for randomized unit test generation. FSS allowed us to cut down on the number of gene types we used, without a noticeable loss of coverage. Since each gene type corresponded to runtime cost, this also allowed us to cut down on the time and memory taken for the algorithm. [5]

James H. Andrews, Tim Menzies and Felix C. H. Li [2011], they described Nighthawk, a system which uses a genetic algorithm (GA), that is used to find parameters for randomized unit testing that optimize test coverage. They reduced GA with Feature sub set selection tool, achieves as same results as the full system, but with 10% of the time. They have shown that Nighthawk is able to achieve high coverage. This technique achieves a good test coverage and produce optimal result. The optimization of the parameter is better while comparing with the previous works. The drawback in this technique is that the coverage criteria are not is not good. The time taken to execute the parameters is also a drawback. [7]

Designing GA is complex issue with its large set of population size. The process of iteration done sequentially so, the process of finding an optimal solution becomes too complex and time consuming, there is no high test case coverage. To simplify this process of reducing the complexity and time conception, PSO algorithm is introduced for randomized unit that achieves high coverage and reduces the time.

III. GENETIC ALGORITHM FOR RANDOMIZED UNIT TESTING

Randomized unit testing is the process of unit testing where there is some randomization takes place to get a successful chromosome (set of test data). A main concept in randomized unit testing is reuse of values. Genetic algorithm approach is used to search and find for a good value parameter to search space [11]. Genetic algorithm is used to making decisions about the features of selecting, crossover and mutating [14]. It can defeat purely random search in finding solutions to complex problems. The Genetic algorithm performs the usual candidate solution evaluation steps (fitness selection, mutation, and recombination). The performance of genetic algorithm is in the sequential process. Genetic algorithm uses Feature subset selection (FSS) tool is used to reduce the test case, which prunes the GA system representation [6][8]. It finds and removes unwanted information in the search control. The process of genetic algorithm is to find parameters for randomized unit testing. Thus the system optimizes the test coverage under randomized unit testing.

GA for randomized unit testing are used to get an optimal test data, but the optimization of testing process is not optimal and minimum of test case generation and less test data coverage in test case, system does not consider the fitness value for randomized unit testing it generates the random selection of software units.

a. Due to the iteration process comes too complex and time consuming.
b. It does not cover of all test cases in the system and values of the parameters are not optimal, thus it produces less coverage of test case.
c. It does not minimize the test case generation and it will not quickly generate the test case.

IV. PSO ALGORITHM FOR RANDOMIZED UNIT TESTING

Particle swarm optimization (PSO) is like the behavior observed in flocks of birds and schools of fish. In nature, there is a leader who leads the bird or fish group to move, most members of the group follow the leader. In PSO, a potential solution to the considered problem is represented by a particle, similar to the individual in the bird and fish group.

PSO algorithm is used to evaluate the target method solutions and operates the fitness value to the receiver candidate. It is a technique used to explore the search space between the set of target methods and to maximize a set of target method parameters. It uses a number of agents (Particles) that constitute a swarm moving around in the search space to target method .Each target method defines some weight function to the randomized unit testing. Each particle represents a possible solution to the optimization value for RTU. In all iterations the fitness value is calculated by each particle.

Every particle has a best value which is calculated by fitness function which is called P_{best} value. According to the P_{best} value the highest priority value is called as G_{best} value which is an optimal solution. One particle determined by its own best solution found and the global best position discovered by any of the particles in the swarm (population). This means that if a particle discovers a new solution, all the other particles will move closer to it, thus it indicates the region more thoroughly in the process.

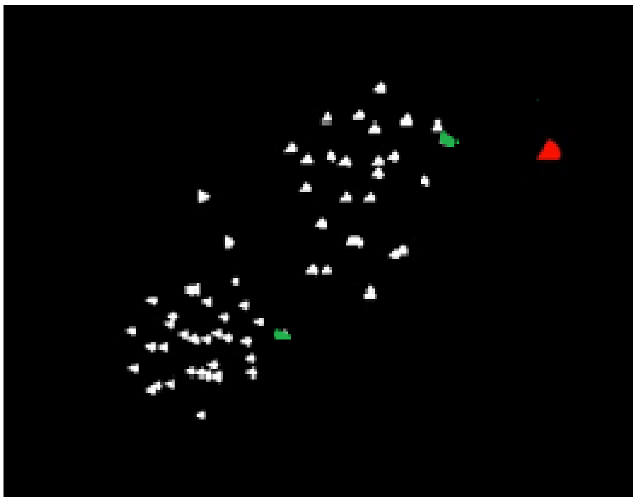

Figure 1: Particle swarm Optimization algorithm

The fig1: shows, each particle have its own leader. Thus all the agents in a particle will follow the leader. Whenever, the iteration takes place the optimal best value will change. According for that value each agent will fly through the search space. The PSO algorithm is used to refer the calling method’s object in target method objective function and arguments of the target method.

Steps in PSO:

a. Initial population (group of test data in the test case).
b. Calculate the fitness value.

Fitness= (No: of coverage point * coverage factor) – No of method call performed

c. Find P_{best} and G_{best} value for each particle.
d. Update velocity and position :

$P_{velocity} = P_{p\text{-}best} + P_{g\text{-}best} + P_{initial\ velocity\ value}$
$P_{position} = P_{velocity} + P_{initial\ position}$

e. Repeat the above steps until G_{best} value remains same and that value is a optimal value

The above steps are used find out a better optimal test value and by using this process it produce high test coverage in less time. It solves the parameter space of problem between the target method and objective function of the test data. The result of this approach is the test coverage time is less when compared with the GA with FSS.

V. RESULTS AND DISCUSSIONS

In this paper proposed optimization algorithm PSO for randomized unit testing. The representativeness of the units under test is a complex issue to external validity. The process of assigning optimization search algorithm for a particular test case of an application shows the effective test coverage. While comparing with the existing algorithm, PSO algorithm gives a best result. The lines of code covered by the test cases, as measured by the number of code covered are the ratio of lines covered.

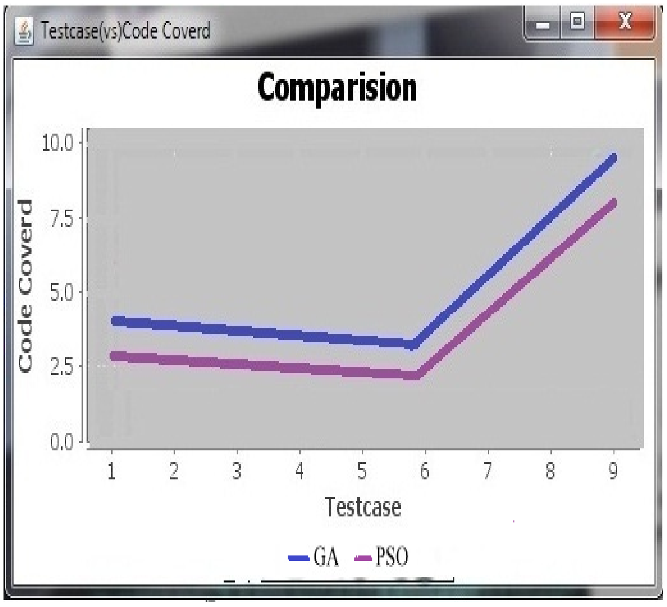

Figure 2: The Test case Vs Code Coverage

This figure shows the coverage of the test case for the methods. Thus, the performance increases. Coverage of number of test case is high, when the code coverage or the result of software is high. Then the performance of coverage of test case is also becoming high.

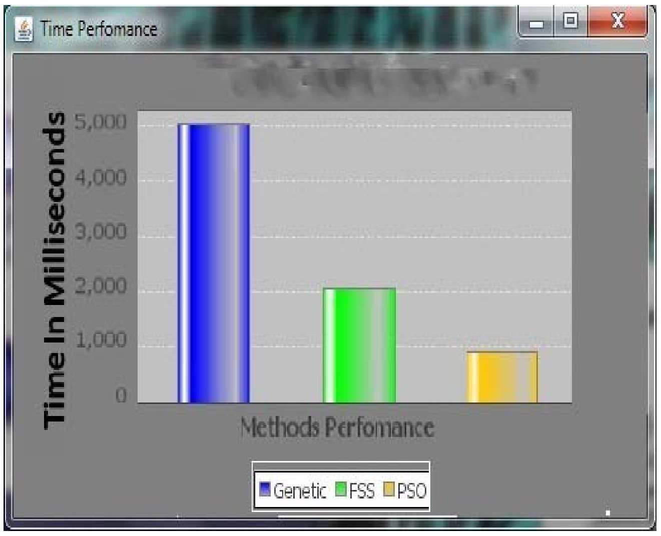

Figure: 3 Time Vs Methods

This fig.3 shows the time coverage for each method. Time is calculated by the time taken for each method to execute the test case of an application. Measuring the time performance of proposed algorithm (PSO) by using the chart, the performance is compared between genetic algorithm, genetic algorithm with FSS tool & PSO.

Here, the performance time of all the methods calculated by milliseconds.

Time taken to execute test case of an application by:

a. GA - 5048 milliseconds,

b. FSS - 2070 milliseconds and

c. PSO - 920 milliseconds.

Thus the time taken by PSO is very less while comparing with GA.

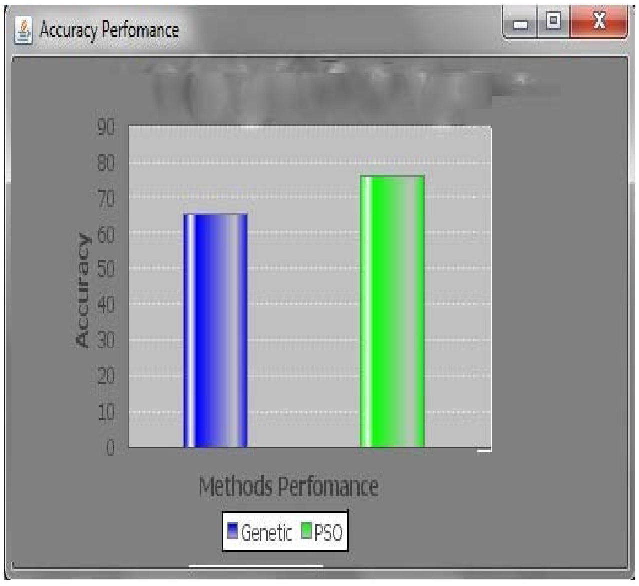

Figure 4: Method Vs Accuracy

This figure shows that the accuracy performance of proposed algorithm (PSO) by using the chart, the performance is compared between genetic algorithm, genetic algorithm with FSS tool and PSO. The accuracy of GA is 66.54%; PSO is 77.87% .Thus PSO increases accuracy up to 10%.

Finally the performance of the system is high, with in less time to coverage of the test case, when the code coverage of the result of the software is high and also the proposed system performance is high when comparing to the existing approaches.

VI. CONCLUSION

PSO automatically derives good parameter values for randomized unit test algorithm and it is able to achieve high coverage and also the ability to generate many new high-coverage test cases quickly and easily. This paper proposed an algorithm PSO which is used to evaluate the target method solutions and operates the fitness value to the receiver candidate for Randomized Unit Testing. It solves the parameter space of problem between the target method and objective function of the test data. It produce an optimal test value for RTU, it achieves high test case coverage with in less time and produces better accuracy. In future, the proposed algorithm PSO can be enhanced by giving a weighted mean value for each particle for randomized unit testing.

VII. REFERENCE

[1]. James H. Andrews, Susmita Haldar, Yong Lei and Felix Chun Hang Li.”Tool Support for Randomized Unit Testing”, July 20, 2006 ACM.

[2]. James M, Marek , Howard Z, ” Automatic Test data generation using genetic algorithm and program dependence graph”, Information and Software Technology Volume 48, Issue 7, July 2006, Pages 586–605.

[3]. Changhe Li, Shengxiang Yang, and Trung Thanh guyen “ A Self-Learning Particle Swarm Optimizer for Global Optimization Problems “ ieee transactions on systems, man,

and cybernetics—part b: ernetics, vol. 42, no. 3, june 2012 627.

[4]. James H. Andrews and Felix C. H. Li" Nighthawk: A Two-Level Genetic-Random Unit Test Data Generator", ASE'07, November 4–9, 2007, Atlanta, Georgia, USA. ACM 978-1-59593-882-4/07/0011.

[5]. James H. Andrews, Tim Menzies, and Felix C. H. Li,"Controlling Randomized Unit Testing With Genetic Algorithms",September 27, 2009.

[6]. James H. Andrews and Tim Menzies, "On the Value of Combining Feature Subset Selection with Genetic Algorithms: Faster Learning of Coverage Models", University of Western Ontario, Published by ACM 2009 Article.

[7]. James H. Andrews, Tim Menzies and Felix C. H. Li ," Genetic Algorithms for Randomized Unit Testing", Ieee transactions on software engineering, vol. 1, no. 1, January 2011.

[8]. Jahnavi K and Basha P, "Dynamic Optimization of Genetic Algorithms for Randomized Unit testing by using FSS ", International Conference on Computing and Control Engineering (ICCCE 2012), 12 & 13 April, 2012.

[9]. Kumar A, Alzabidi M and A.D. Shaligram ,"Automatic Software Structural Testing by Using Evolutionary Algorithms for Test Data Generations", IJCSNS International Journal of Computer Science and Network Security, VOL.9 No.4, April 2009.

[10]. Serkan Kiranyaz, Turker Ince, Alper Yildirim, and Moncef Gabbouj "Fractional Particle Swarm Optimization inMultidimensional Search Space" Ieee transactions on systems, man, and cybernetics—part b: cybernetics, vol. 40, no. 2, april 2010.

[11]. Shenga Z, Ying Zhang, HZ and Qingguan H, "Automatic path test data generation based on GA-PSO", Intelligent Computing and Intelligent Systems (ICIS), 2010 IEEE International Conference on 29-31 Oct. 2010 Volume**:** 1 Page(s): 142 – 146.

[12]. Tan, R. P. and Edwards, S, "Evaluating Automated Unit Testing in Sulu", Software Testing Verification an Validation 2008 is International Conference on Date of Conference: 9-11 April 2008 Page(s): 62 - 71.

[13]. Xuan P and Lu L , "A new approach for session-based test case generation by GA", Communication Software and Networks (ICCSN), 2011 IEEE 3rd International Conference on Date of Conference: 27-29 May 2011 Page(s): 91 - 96 .

[14]. Nirmal Kumar G and Mukesh Kumar R," Using Genetic Algorithm for Unit Testing Of Object Oriented Software", First International Conference on Emerging Trends in Engineering and Technology 978-0-7695-3267-7/08 $25.00 © 2008 IEEE DOI 10.1109/ICETET.2008.137.