ISSN: 2278 – 909X
International Journal of Advanced Research in Electronics and Communication Engineering (IJARECE)
Volume 6, Issue 7, July 2017

Test Case Optimization for Regression Testing Based Hybrid PSABC-Firefly Technique

Renuka singh[1] (M.tech scholar)

Email id-rssinghrenuka@gmail.com

Sania Thomas[2] (Assistant Professor)

Computer science engineering

Subharti Institute of Technology and Engineering (Meerut, India)

ABSTRACT

The software preservation phase in a Software Development Life Cycle involves regression testing to be done for retest the existing test suite whenever any alterations are done to the software test case optimization in the specification-based on requirement severity score and inter-case dependency, and to improve the test case preparation done the application of meta-heuristics. We use the Breshman technique with PSABC (particle swarm artificial bee colony)-Firefly hybrid approachto reduce the complexity of code & traverse with each line of code, space symbol, null character, & related user defined comments. Simulation results using MATLAB has also been done which gives near optimal solutions using fitness function of soft computing approach. And its test case optimization has been researched extensively in the past decade.

Keywords*: Software Testing, SDLC, Firefly technique, inter test case etc.*

I.INTRODUCTION

A software development business reason method that must to be absorbed on calculating delivery for a connected order might end up burdened with boilerplate cipher dedicated to making an audit trail, validating the user, informing the UI that a alteration has happened, and performing its task on a employee thread.

1.1. An enormous amount of code is duplicated,

A software program are often derivative and stuck into novel methods, which can chief to duplication of germs and upsurges the trouble of refactoring the cross-cutting anxieties. In other arguments, OOP requests end up obligating one of the actual evils that OOP was intended to prevent: code repetition.

1.2. Particle swarmoptimization

In PSO (Particle Swarm Optimization), a search space has been explored for problems and parameters are identified which maximizes aim of that given problem. In PSO, "n" particles and position of each particle stands a potential solution of given problem. In 1995, James Kennedy and Russell C. Eberhart [11] gave this technique, by drawing two individual concepts of PSO:

1. By observing swarming habits of special kinds of animals (flock of birds, school of fishes), derived the thought of swarm intelligence and,
2. Evolutionary Computing Algorithms.

The whole procedure begins by initializing population (a class of particles) that promises a potential solution and thence, optimal solution searched by position and velocity updation of particles [12]. Each particle in a given population has been assigned position and velocity of its own. The updation and modification is carried out to acquire "best" or optimal solution for the stated problem. With each iteration, values that are "best" for particle (individually) are identified based on: the value that is best achieved so far by a particle (pbest), whereas

the best value incurred by some other particle with in population (gbest) and the best value achieved by immediate neighbors (local best) of a particle (lbest).To support the above principle following equations represents modification in velocity of particle which further causes position updation [13]:

$V_{ik}+1 = w*V_{ik} + c_1\ rand_1\ (\ldots)\ *(pbest_i-s_{ik}) + c_2\ rand_2\ (\ldots)\ *\ (gbest-s_{ik})$ (1)

Where, v_{ik} : velocity of agent i at iteration
w: weighing function,
c_j : weighting factor (uniformly distributed random number between 0 and 1),
s_{ik} : current position of agent i at iteration k,
$pbest_i$:pbest of agent i,
gbest: gbest of the group.
And, the weighting function is calculated as:
w = wMax-[(wMax-wMin) x iter]/maxIter
(2)
where, wMax= initial weight,
wMin = final weight,
maxIter = maximum iteration number,
iter = current iteration number.
$s_{ik}+1 = s_{ik} + V_{ik}+1$
(3)
In whole, the PSO is an optimization approach that updates and validates each step. The process of updationcomes from solution space only. A particle updates itself depending on the global best or local best. The random variable chosen is not generated but formulated by analyzing the population's position & velocity and particle's position & velocity.

1.3. Artificial Bee Colony (ABC)

ABC algorithm [14] is a swarm-based metaheuristic algorithm simulating the behavior of honeybees. When applied to ELD problem, the solution produced by the algorithm is represented by the location of source of nectar while the amount of nectar represents the quality (fitness) of the solution. Employee bees fly around in search of source of nectar (representing trial in a search space) and select their preferred source of nectar based on their experience. Once search is completed, they share their findings (source) with the onlooker bees waiting in the hive. The onlooker bees then make a probabilistic selection of new source of nectar based on the information received from the employee bees. Only if the amount of nectar of the new source is higher than that of the old one, the onlookers choose the new position. If the quality of solution is not improved by a predetermined number of trials (finding sources with higher nectar), then the scout bees fly to choose new source randomly abandoning the old source. If the abandoned source is x_{pq}, q ϵ {1,2. . .,D}, where p is a solution and q is a randomly chosen index, then a new
nectar source chosen by a bee is given by:

$x_{pq}=xq_{\min}+ rand(0,1) * (xqmax–x_{qmin})$ (8)

Here, x_q max and x_q *min* are the maximum and minimum limits of the ELD parameter to be optimized. Each solution in the ELD solution space is a vector, D being the number of optimization parameters [15].

1.4. Firefly Algorithm (FA)

Firefly algorithm is a novel metaheuristic optimization algorithm and it has been applied successfully for ELD problem by Sudhakara et al. [16] and for Economic Emission Dispatch (EED) problem by Apostolopoulos et al. [17]. The algorithm is based on the social flashing behavior of fireflies; two important issues are the variation of light intensity (associated with the objective function) and the formulation of the attractiveness. Since a firefly's attractiveness (β)is proportional to the light intensity (I) seen by adjacent fireflies and I varies as: $I(r)= I_0e^{-\gamma r2}$, the attractiveness β(r) at a distance r is determined by: $\beta(r) =\beta_0e^{-\gamma r2}$. The movement of a firefly i attracted to another firefly j is determined by:

$x_{i+1} = x_i + \beta_0e^{-\gamma r2}\ (x_j-x_i) +\alpha(rand-1/2)$ (9)

Here, rand is a random number generator distributed uniformly in [0, 1] space and α is a randomization parameter. The third term in Eq. (9) is used to represent random movement of fireflies in case there are no brighter fireflies to attract the movement.
The FA has many similarities with other swarm intelligence-based algorithms, e.g., PSO, ACO, and ABC but from implementation point of view it is simpler than the others.

II.LITRATURE REVIEW

Robinson et al. (2002)[1] proposed two hybrid algorithms; GA-PSO and PSO-GA for solving a particular electromagneticapplication of profiled corrugated horned antenna. In GA-PSO algorithm, GA was used to generate the initial populationfor

PSO, while in the second algorithm PSO-GA, PSO generates an initial population for GA. His results showed that the PSOGAhybrid algorithm outperforms the GA-PSO version as well as simple PSO and GA versions.

Grimaldi et al. [2] proposed a hybrid technique combining GA and PSO called genetical swarm optimization (GSO) forsolving combinatorial optimization problems. They applied their algorithm for solving an electromagnetic optimizationproblem. In GSO, the population is divided into two parts and is evolved with these two techniques (GA and PSO) in everyiteration. The populations are then recombined in the updated population, that is again divided randomly into two parts inthe next iteration for another run of genetic or particle swarm operators. They defined a new parameter HC (or HybridizationConstant) that expresses the percentage of population that is evolved with GA in every iteration: so HC = 0 implies that theprocedure is pure PSO (the whole population is updated according to PSO operators) and HC = 1 implied that pure GA is beingfollowed. However, 0 < HC < 1 means that the corresponding percentage of the population is developed by GA and the restwith PSO.

Settles and Soule [3] introduced a GA and PSO hybrid called the breeding swarm (BS) algorithm. The BS algorithmcombines the standard velocity and position update rules of PSOs with the ideas of selection, crossover and mutation fromGAs. They also included an additional parameter called the breeding ratio to determine the proportion of the populationwhich undergoes breeding in the current generation. Their algorithm consisted of four main steps; in the first step, n bestindividuals are copied into a temporary population, in the second step some individuals are selected to undergo the velocityand position update rules of PSO. In the next step the remaining individuals are selected by some selection criteria for crossoverand mutation criteria. Finally the temporary population is copied into the working population and the fitness is evaluated.This process is repeated till the optimum is obtained. The BS algorithm was validated on four unconstrained scalableoptimization problems for different dimensions.

Jian and Chen [4] introduced a PSO hybrid with the GA recombination operator and dynamic linkage discovery calledPSO-RDL. They assumed that the relation between different dimensions is dynamically changed along with the search process.Thus, the linkage configuration should be updated accordingly. Instead of incorporating extra artificial criteria for linkageadaptation, they entrusted the task to the mechanism of natural selection. The idea is to update the linkage configurationaccording to the fitness feedback. They applied their algorithm for solving 25 unconstrained test problems with varying degreesof complexities.

In the PSO-GA based hybrid evolutionary algorithm (HEA) of **Yang et al. [5],** evolution process is divided into two stages.The first stage is similar to PSO where the particle flies in hyperspace and adjusts its velocity by following particles with betterfitness according to flying experience of itself and its neighbors. The second stage is similar to GA where genetic operatorsof selection, reproduction, crossover, and mutation are exerted on particles at predetermined probability. By combination ofPSO and GA, evolution process is accelerated by flying behavior and population diversity is enhanced by genetic mechanism.They used a single point crossover, Gaussian mutation and Roulette wheel for selection process. They applied their algorithmfor solving 3 unconstrained as well as 3 constrained optimization problems.

Anurag Gupta et al, 2012 [6] uses PSO on the combined economic and emission dispatch problem. It combines the two objectives into one using the price penalty function. It shows a better advantage in terms of cost, fast convergence, and less computational time than other heuristic methods like GA and dynamic programming. Also PSO gives efficiently high quality solutions with more stable convergence characteristics than the other heuristic methods afore mentioned.

Lakshmi A. Devi et al, 2008 [7] uses the evolutionary programming method on the combined economic and emission dispatch problem. This paper proposes the use of the lambda in the evolutionary algorithm with the reason being that it makes the coding of the chromosomes independent on the number of units. Notably PSO generates a lower fuel

cost and emission release but sometimes has a higher computational time than GA.

M. R. Alrashidi et al, 2008 [8] on the impact of loading conditions on the emission and economic dispatch problem uses weighting functions on the double objective of emission and fuel cost. It provides a simple way of addressing the equality constraint. The rule guiding the application of the weights to the objectives is not explicitly shown. Also this method is not applied to the CEED rather it optimizes the objectives independently.

Gaurav Prasad et al, 2011 [9] applies a new technique called Artificial Bee Colony method (ABC) the economic load dispatch problem. In comparison to other heuristic methods it shows highly superior features like quality of solution, stable convergence characteristics and good computational efficiency. It does not consider the environmental or emission dispatch problem.

Sonmez et al, 2011 [10] applies the Artificial Bee Colony method to solve the multi-objective economic and environmental dispatch problem using the penalty factor approach. It is superior in comparison to the other heuristic methods and more efficient. In this research work, exploration of the area of hybridizing PSO and the Artificial Bee Colony method and studies of its behavior in comparison with the other methods using the combined emission and economic dispatch problem was be done.

III.PROBLEM STATEMENT

Major changes are carried out to complete the encapsulation of a candidate aspect or to fix problems in the source code so as to allow correct compilation. When the application of an aspect refactoring is not sufficient to encapsulate a language element of a candidate aspect this situation can be solved in two ways.The order in which these are selected can produce problems of compilation or encapsulation in the resultant system. This order depends on the structure of the candidate aspects. While in practice this would be possible, the reason why we can ‘not use association rules to identify the fragments of code to be migrated and to recommend additional restructurings is that the association rules technique is not flexible enough to identify new problems without regenerating the association rules through the association rules algorithm.

IV. PROPOSED IMPLEMENTATION

Start => MATLAB code => use Breshman line code=>Reduce visible fault => Testing outcome on command window => Test sequence generation => Finding faults=> Correcting faults=> Get graph result of crosscutting matrices=> Test case optimize using PSABC-Fireflies

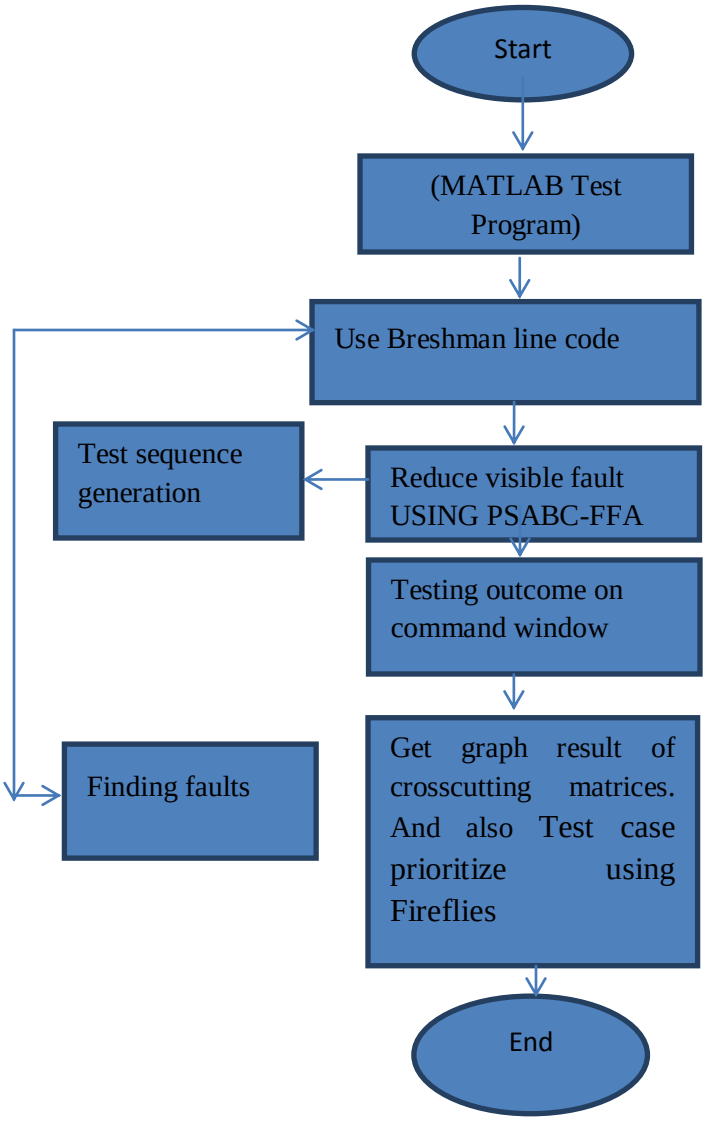

Figure 4.1 Proposed flow chart

V.RESULT

Test case tasks & initial testing more fix solution expensive & manipulations becomes with each code testing in different type of languages. This will set the position of the scope of the early tricky while conserving a better performance & thus foremost to earlier & improved convergence. Code coverage &Breshman line are built in model of usual collection & novel method categories out original testing so that individual higher result are reserved with soft computing technique such as Proposed firefly with PSABC.

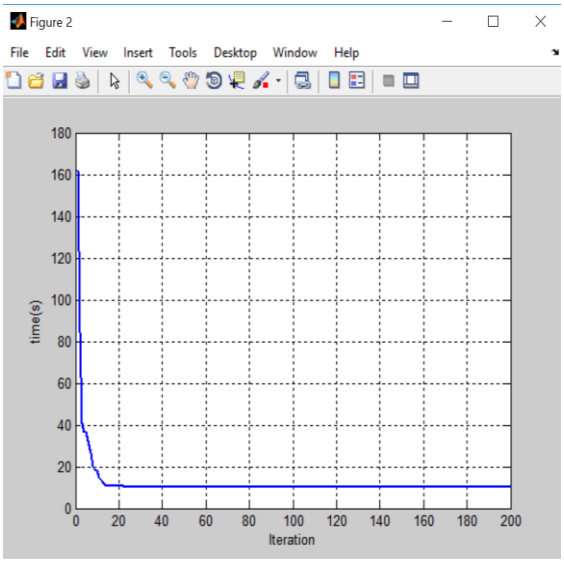

Figure 5.1: Code Execution time for proposed technique quality for line of code improvement.

In above figure a feature for all homogeneity totient function & its code P, & describe the next metrics terms of all function:

5.1.**Structure for execution time value**

In all related metrics & the amount of programs that are crosscut by all modules information & relation of feature are fix in a feature.

5.2.Finite state of ACD (Advice Crosscutting Degree)

The total no of classes that comprehensive idea for crosscut completely by the fragments of information in a feature & light weighted position.

ACD(f,P)=count(sclasses(shadows(pointcuts(advices (aspects(f))),P))) HQ.The Homogeneity Quotient are the technique that having separation of the information crosscutting degree (ACD)& defined by its feature crosscutting degree (FECRDE): HQ(f,P) = ACD (f,P)/FECRDE(f,P) if FECRDE(f,P)!=0 = 0 otherwise

Working PHQ (Program Homogeneity Quotient) for finding NOF:

It is similar to the outline of homogeneity quotients for features of total code lines of any object oriented language which, divided by the number of features for all vector matrix (condition):

NOF. PHQ(P) = sum (for each(P, λg. HQ (g,P)))/NOF(P)

5.3. SOME DEFINITION OF THE RESULT WORK

sloc Counts number source lines of code.SL = sloc(FILE) returns the line count for FILE. If there are multiple functions in one file, sub-functions are not counted separately, but rather together.

The following lines are not counted as a line of code:

(1) The "function" line

(2) A line that is continued from the previous line -

(3) A comment line, a line that starts with --> % or a line that is part of a block comment (%{...%})

(4) A blank line

Note: If more than one statement is on the line, it counts that as one line of code. For instance the following:

minx = 32; maxx = 100; is considered to be one line of code. Also, if the creation of a matrix (condition) is continued onto several line without the use of '...', sloc will deem that as separate lines of code. Using '...' will "tie" the lines together.

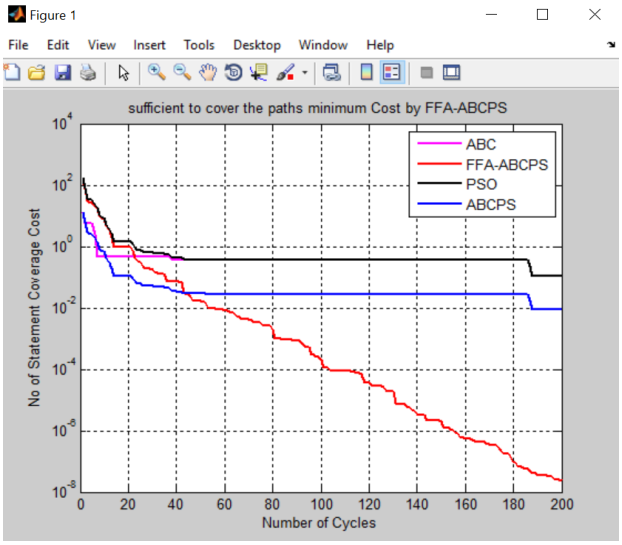

Figure 5.2: No of Statement Coverage for proposed approach with each cycle

In figure 5.2 we have find out the best cycle function on the various iteration number. Table is shown in below.

VI.CONCLUSION

To fix deployment tools that can be used at different concept levels, authorizing designers to measure modularity and conclude quality properties at early stages of development although the modification for model-driven software engineering ahead momentum, the valuation of abstract becomes more significant structure. detect the fault in in dynamic soft computing approach firefly that can be enable cost reduction and minimization time complexity and counting the degree of HQ value for each line of codes are general and they are not secure to fix positioning tools that can be used at changed concept levels, authorizing designers to measure modularity and conclude excellence belongings at early phases of expansion although the modification for model-driven software manufacturing fast momentum, the valuation of abstract mockups becomes more significant structure.

REFERANCES

[1] J. Robinson, S. Sinton, Y.R. Samii, Particle swarm, genetic algorithm, and their hybrids: optimization of a profiled corrugated horn antenna, in:Proceedings of the IEEE International Symposium in Antennas and Propagation Society, 2002, pp. 314–317.

[2] E.A. Grimaldi, F. Grimacia, M. Mussetta, P. Pirinoli, R.E. Zich, A new hybrid genetical – swarm algorithm for electromagnetic optimization, in:Proceedings of International Conference on Computational Electromagnetics and its Applications, Beijing, China, 2004, pp. 157–160.

[3] M. Settles, T. Soule, Breeding swarms: a GA/PSO hybrid, Proceedings of Genetic and Evolutionary Computation Conference (2005) 161–168.

[4] M. Jian, Y. Chen, Introducing recombination with dynamic linkage discovery to particle swarm optimization, Proceedings of the Genetic andEvolutionary Computation Conference (2006) 85–86.

[5] Yang, Y. Chen, Z. Zhao, A hybrid evolutionary algorithm by combination of PSO and GA for unconstrained and constrained optimization problems, in:Proceedings of the IEEE International Conference on Control and Automation, 2007, pp. 166–170.

[6] Anurag Gupta, K.K. Swarnka, Dr. S. Wadhwani, Dr. A.K. Wadhwani, " combined Economic and Emission Dispatch problem of thermal generating units using Particle Swarm Optimization", International Journal of Scientific and Research Publications, Vol 2, Issue 7, July 2012

[7] A. Lakshmi Devi , O. Vamsi Krishna, " combined Economic and Emission dispatch using evolutionary algorithms- A case study", ARPN Journal of engineering and applied sciences, Vol. 3, no 6, December 2008.

[8] M.R. Alrashidi, M.E. El-Hawary, " impact of loading conditions on the emission-economic dispatch", World academy of science engineering and technology 39, 2008

[9] Gaurav Prasad Dixit, Hari Mohan Dubey, ManjareePandit, B.K. Panigrahi, " Economic load

dispatch using Artificial Bee Colony Optimization", International Journal of Advances in Electronics Engineering, 2011

[10] Y. Sonmez, " Multi-objective environmental/ economic dispatch solution with penalty factor using Artificial Bee Colony algorithm" , Scientific Research and Essays, Vol. 6 (13), pp 2824-2831, 4th July 2011

[11] J. Kennedy and R. C. Eberhart., Swarm Intelligence, Morgan Kaufmann Publishers, San Francisco, (2001).

[12] P.J. Angeline, "Using Selection to Improve Particle Swarm Optimization", IEEE int. Conf., , pp. 84-89, 1998.

[13] M. Mitchell. An Introduction to Genetic Algorithms . MIT Press, Cambridge, MA, 1998.

[14] Dorigo M (1992) Optimization, learning and natural algorithms. PhD Thesis, Dipartimento di Elettronica, Politecnico di Milano, Italy

[15] Sonmez Y (2011) Multi-objective environmental/ economic dispatch solution with penalty factor using Artificial Bee Colony algorithm. Sci Res Essays 6(13):2824–2831

[16] Sudhakara Reddy K, Damodar Reddy M (2012) Economic load dispatch using firefly algorithm. Int J Eng Res Appl (IJERA) 2:2325–2330

[17] Apostolopoulos T, Vlachos A (2011) Application of the firefly algorithm for solving the economic emissions load dispatch problem. Int J Comb 1(3):1–23