International Journal of Engineering &Technology, 7 (1.7) (2018) 210-216

International Journal of Engineering & Technology

Website: www.sciencepubco.com/index.php/IJET

Research paper

Particle swarm optimization to produce optimal solution

C. Saranya Jothi[1*],V.Usha[2] , R. Nithya[3]

[1,2]Assistant Professor, Department of Computer Science and Engineering, School of Computing,
Vel Tech Rangarajan Dr. Sagunthala R&D Institute of Science and Technology, Avadi, Chennai-62, TamilNadu, India.
**Email: saranyajothi22@gmail.com*

Abstract

Search-Based Software Testing is the utilization of a meta-heuristic improving scan procedure for the programmed age of test information. Particle Swarm Optimization (PSO) is one of those technique. It can be used in testing to generate optimal test data solution based on an objective function that utilises branch coverage as criteria. Software under test is given as input to the algorithm. The problem becomes a minimization problem where our aim is to obtain test data with minimum fitness value. This is called the ideal test information for the given programming under test. PSO algorithm is found to outperform most of the optimization techniques by finding least value for fitness function. The algorithm is applied to various software under tests and checked whether it can produce optimal test data. Parameters are tuned so as to obtain better results.

Keywords: *Search Base Testing, PSO, Optimization, Swarm Particle.*

1. Introduction

Automation of testing is needed for huge scale exhaustive testing to be down to earth and adaptable. The robotized age of test information presents different challenges. The basic issue of observing perfect game plan is known to be undecidable, So examine has focused on frameworks that hope to perceive close perfect test data in sensible time.

1.1 Inquiry Based Software Testing

Hunt Based Software Engineering is the field of treating soft-ware engineering as a search problem, i.e. using various search algorithms to generate the data defining the solution to the engineering problem. In particular, software testing is an area which has seen much progress using this approach. This special form of SBSE is commonly called Search based Software Test Data Generation (SBSTDG). Test information age is an undecidable issue. There exists two primary methodologies for testing: White Box (auxiliary) or Black Box (utilitarian) testing. Basic testing checks the conduct of the product by getting to its code. Then again Functional testing explores whether the product functionalities are carrying on of course with no learning about the code . The two sorts of testing depends on age and execution of test suite, that is an arrangement of experiments.

1.2 Control Flow Graph

The control stream chart (CFG) is a portrayal of the control structures of a program (or other unit of code). In the description taken from Tracey et al., the nodes in this directed graph correspond to blocks of code, of which all or none are executed. The edges are possible transfers of control, and in the case of more than one edge out from a node, a branch predicate will decide between them. The branch predicate is a control flow statement, such as if (a == 1). A test ampleness paradigm is some measure or set of measures that a test needs to satisfy so as to be esteemed fruitful. Figure 1.1 shows a sample CFG.

1.3 Local Search

Nearby inquiry utilizes the wellness capacity to assess conceivable moves inside the interest space from a lone current course of action point until the point when an area perfect is come to. Nearby pursuit may wind up plainly caught in neighborhood optima inside the arrangement space, however can be much more productive for less difficult inquiry issues. In programming designing terms, this sets up an evident occasion of the exemplary exchange off amongst proficiency and viability; one may anticipate that world-wide hunt will accomplish preferable branch scope over nearby inquiry, however at the cost of more prominent computational exertion.

Hill Climbing Hill Climbing Algorithm Software Under Test(SUT)will be given test data so that control flow will reach the desired target node. SBST aims to find test data with least fitness value which is calculated using an objective function [Koral] Hill Climbing Algorithm will start from initial random population(test data). Fitness value corresponding to this initial test data is calculated. Test data is chosen in such a way that the control flow will reach the selected target node. Algorithm proceeds by searching the neighbourhood of initial test case. Fitness value is calculated each time. On the off chance that the new wellness esteem is not as much as present, at that point the new wellness esteem is acknowledged. If it is higher, then the new fitness is rejected and algorithm assumes that it cannot find a lesser value thereafter. Then the algorithm will exit. The least value found by hill climbing may not be optimal. This is called getting stuck in local optima. Figure 1 shows the plot between fitness value and number of iterations.

Copyright © 2016 c.saranya jothi. This is an open access article distributed under the Creative Commons Attribution License, which permits unrestricted use, distribution, and reproduction in any medium, provided the original work is properly cited.

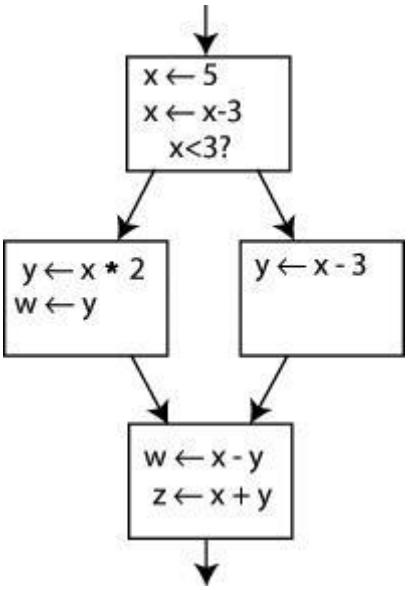


Fig 1: Control Flow Graph

1.4 Global Search

Worldwide hunt includes an accumulation (a populace) of applicant arrangements that advance after some time, taking into account a wide inspecting of the inquiry space. Worldwide hunt procedures expect to defeat the issue of nearby ideal in the pursuit space and can in this way discover all the more all inclusive ideal arrangements.

1.5 Search-Based Testing Algorithms

Typically when searching for test input to a given program, an objective function is defined, which computes how well a candidate solution solves the search problem. One search technique uses what is known as the branch distance in this objective function, a measure of how far a control flow statement such as an if-then-else-statement is from evaluating to either true or false. The software being tested is typically instrumented, meaning code is inserted in order to monitor the execution of the software. To compute the branch distances, some parts of the code may be (temporarily) transformed as well.
Metaheuristic search techniques Metaheuristic seek procedures are abnormal state systems which use heuristics so as to discover answers for combinatorial issues at a sensible computational cost. Metaheuris-tic search has been used for generating test data for specific structures in white-box testing to execute a certain structure in a program, finding and triggering flaws in a program, and also to test properties of software, such as execution time.

1.6 Search Algorithms

Normally in a test input search, the algorithms start with random input. This input, and every subsequent proposed solution, are compared to the test adequacy criterion, in order to see if the solution is good enough. An objective function guides this search, by informing the algorithm in question how well the attempt fared. A few common search algorithms are discussed; Arbitrary inquiry, Genetic Algorithm, which is a case of worldwide pursuit, and Hill climbing, which is a case of nearby hunt.
Genetic Algorithms (GA), the variant probably best known out of the larger class of algorithms evolutionary algorithms, use mechanisms of pseudo-natural selection to evolve solutions. A solution is often called an individual, with the parts making up this solution called genes. If the individual is a vector of inputs, one of those inputs would be a gene. One mechanism borrowed from evolution in nature is the recombination of two individuals genes, i.e. the two individuals are breeding. Which genes pass on to off springs can be chosen at random, or some rule may be defined, depending on design and need. Another mechanism is mutation, where a gene has a certain chance of being changed. As the genes consist of inputs to the program, care must be taken to avoid going outside of boundaries the inputs can have in the program.
Hill Climbing (HC) is a nearby inquiry calculation often used in the research literature. From the search space, a starting point is chosen, and the neighborhood in the search space is investigated. If the neighborhood contains a better solution, this is chosen. Then the neighborhood seek proceeds, until the point when no better arrangement can be found. The solution is then an optimum, either local or global. The name ”Hill climbing” comes from visualizing the search space as a landscape, where the peaks represent good values for the objective function for the particular input in the search space, and the valleys represent bad values. The neighborhood search thus represents climbing these hills.
Fitness Function and Branch Distance Role of the fitness function The fitness function (in general optimization problems called objective function) is what informs the search algorithm of the quality of a solution, and hence possible improvement or worsening of this quality in successive proposed solutions . This also holds true when searching for and generating test input data. The fitness function computes a value representing how well the solution is fulfilling the test adequacy criterion.
Approach level The approximation level, also approach level, is a measure of how a long way from executing the objective branch the present arrangement hopeful came. To execute an objective branch the program must execute a certain set of branches leading up to it. Whenever the test inputs make the flow of control divert from these branches, the target cannot be reached. The number of predicates in this set still not executed is the approach level.

2. System Overview

Particle Swarm Optimization
Particle swarm optimizations require several parameters including the following:

- Initial Particles.
- Velocity of Particles.
- Direction of Particles.
- Number of Iterations.
- Fitness Function.

2.1. The Initial Particles

Particles are assumed to be moving in the hyper dimensional space with certain velocity. Each particles aims to reach a better position which is determined by the objective function. Particle positions and velocities are initialised randomly.

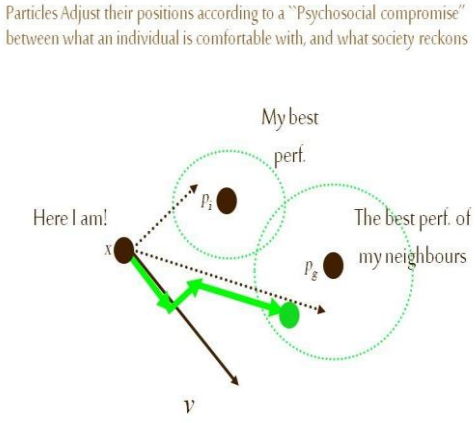


Fig:2 Particle Motion

2.2 Velocity of Particles

Each particle is randomly assigned with a velocity. Position of particle after some time t is determined by the initial position and velocity.

2.3 Direction of Particles

Movement of each particle will be determined by the resultant of its inertia(Initial direction of movement), particles best position (pbest) and overall best position found so far (gbest). The influence of these factors on particle motion is shown in Figure 2.

2.4 Number of Iterations

Particles will keep on moving as the algorithm proceeds. As the iterations move on, the particle tend to converge to a common position. This is the optimal solution. Stopping criterion for the algorithm may be when it reach the optimal solution or when it reaches the iteration limit. The particles overall best position at the last iteration is treated as optimal solution.

2.5 Fitness Function

It is expected that the branch predicates are basic social articulations (disparities and correspondences). That is, all branch predicates are of the accompanying structure: El OP E2.The branch remove figuring's relating to each if predicate is specified in Table 1. F is a genuine esteemed capacity, alluded to as a branch work, which is

1. positive when a branch predicate is false or
2. negative when the branch predicate is valid.

Clearly F is really a component of program input x. Emblematic assessment can be utilized to discover express portrayal of the F (x) regarding the info factors. [Koral 1990] The wellness work consolidates a measure known as the approach level with the branch remove

Table 1: Branch Distance Calculations

Branch Predicate	Branch Function
E1>E2	E2 E1
E1>=E2	E2 E1
E1<E2	E1=E2
E1<=E2	E1-E2
E1=E2	abs(E1-E2)
E1! =E2	abs(E1-E2)
E1;jE2	min(cost(a),cost(b))
E1&&E2	cost(a)+cost(b)
negation(E1)	Negation is moved inwards and propagated over E1

2.6 Approach Level

The approach level is a check of what number of the branches control subordinate hubs were not experienced in the way executed by the information. An information which executes more control subordinate hubs is nearer to achieving the objective as far as the control stream diagram, and in this manner is compensated with a lower approach level.

2.7 PSO in Real Number Space

In the genuine number space, every individual conceivable arrangement can be demonstrated as a molecule that travels through the issue hyperspace. The situation of every molecule is controlled by the vector and its development by the speed of the molecule.

$$v_i^{t+1} = \underbrace{v_i^t}_{\text{inertia}} + \underbrace{c_1 U_1^t (pb_i^t - p_i^t)}_{\text{personal influence}} + \underbrace{c_2 U_2^t (gb^t - p_i^t)}_{\text{social influence}}$$

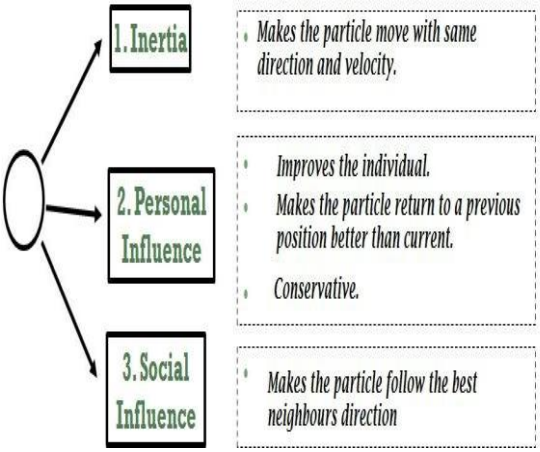


Fig:3 Real number space

The data accessible for every individual depends without anyone else experience (the choices that it has made up until now and the accomplishment of every choice) and the information of the execution of different people in its neighborhood. The factors influencing molecule movement are appeared in Figure 3.2. Since the relative significance of these two elements can differ starting with one choice then onto the next, it is sensible to apply arbitrary weights to each part, and in this way the speed will be dictated.

2.8 PSO Algorithm

1.Initialize the swarm by doling out an irregular position in the prob-lem hyperspace to every molecule.

2.Evaluate the wellness work for every molecule.

3.For every individual molecule, contrast the particles wellness esteem and its . On the off chance that the present esteem is superior to the esteem, at that point set this incentive as the and the present particles position as .

4.Identify the molecule that has the best wellness esteem. The estimation of its wellness work is distinguished as and its situation as .

5.Update the speeds and places of the considerable number of particles utilizing 1 and 2.

6.Repeat stages 2 to 5 until the point that a halting model is met e.g., most extreme number of emphasess or an adequately decent wellness value.The flowchart for PSO calculation is appeared in Figure 3.3. Figure 3.4 demonstrates the design chart.

3. Modules

- Initialization of particle positions and velocities
- PSO
- Velocity update

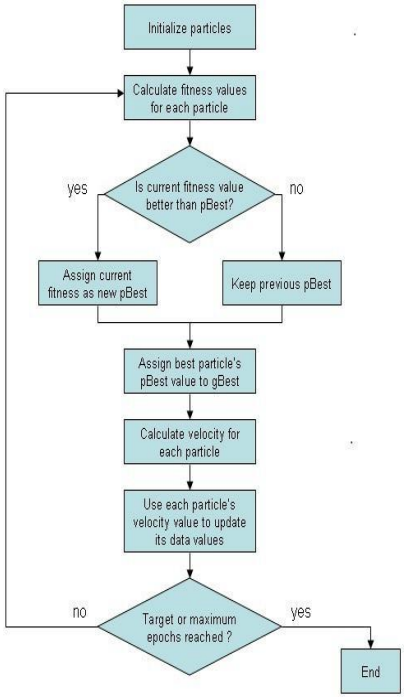


Fig:4 PSO Flowchart

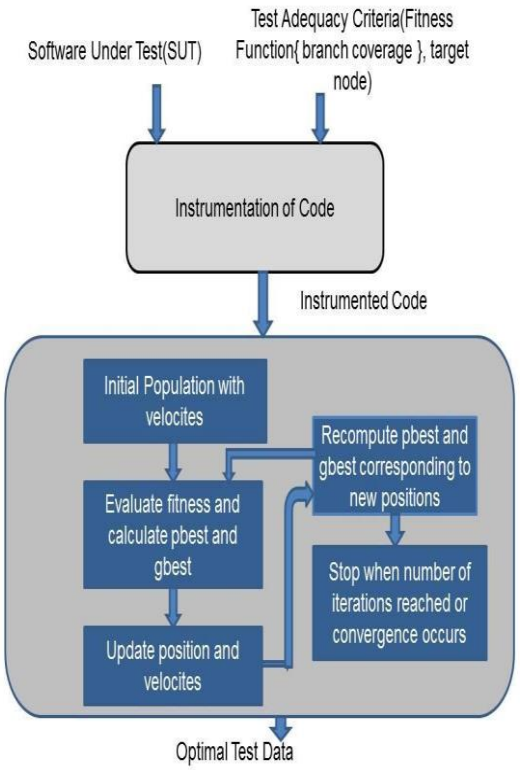


Fig:5 Architecture Diagram

3.1 Instrumentation of Code

Instrumentation is the process of inserting custom code into an existing program, which allows information from the execution of the program to be collected. Code is instrumented to, among other things, be able to see the paths the program takes through the code given a set of input variables. Instrumenting branches consists of inserting a probe as the first statement in the code block representing that branch. In if-then-else predicates, the else-part is not mandatory and will be omitted in the code when it is not needed, i.e. there is no code to be executed.

3.2 PSO

Initialization of Particle positions and velocities. In PSO particles are assumed to move in a hyper dimensional space. Each particle will be having a position and velocity corresponding to it. In software testing we assume that the particles are test data. The position of the particles correspond to the values of test data. Initially particles are assigned random positions and velocities. Fitness Calculation Test data is chosen based on the software under test(SUT). The code is then instrumented and used for the fitness calculation. Each if predicate is treated as a node. One statement is set as target node and fitness is calculated based on this. The test data which is having least value for fitness function is said to be optimal solution.
Each particle keeps track of its personal best and global best. Personal best is the best fitness value that a particle has encountered. Global best is the overall best fitness value of all particles after each iteration.

3.3 Velocity Update

Speed is refreshed after every cycle. It relies upon the pbest and gbest esteems. In the genuine number space, every individual conceivable so-lution can be demonstrated as a molecule that travels through the issue hyperspace. The situation of every molecule is controlled by the vector and its development by the speed of the molecule.

4. Implementation

This chapter explains the details of implementation of modules.

4.1 PSO Applied In Testing

Generation of the test data using Evolutionary Algorithms, such as GA and PSO, for a Software Under Test (SUT) must follow some basic guidelines.

1.Identify the target nodes using branch predicates for the branch coverage with a choice of input test data representation.

2.Based on the target nodes, instrument the code so that it can be used directly for automatic test data generation.

Applying PSO for a Software Under Test(SUT), requires a few parameters to generate test data and it is discussed here. Particle (xi), Swarm, particle position at time, (xi(t)), particle velocity(vi(t)), particle movement, convergence, fitness function, pb, pg , pbest, and gbest are the parameters involved. Particle correspond to a set of input test data in SUT which is of 'n' dimension. Swarm is a collection of the particles in the given SUT, which are randomly initialized and it is referred as initial population. Initially the random values are assigned to the particles position within the range [xmin 29 ,xmax] based on the equation (3), where Xmin is minimal limit of the test data representation and xmax is the maximum limit of the test data representation. Initially at time, t=1, the particle global best position, pg, and their corresponding fitness, gbest are assigned as zero. Initially, at time, t=1, particle velocity is equal to particle position and it is represented in the equation (4). Using the particle position value, the fitness function is evaluated and their corresponding fitness value is obtained. Computation of fitness function . Initially at time, t=1, the particle best position, pb, and their corresponding fitness, pbest, are assigned as initial parti-

cle position and corresponding fitness value respectively. In the next generations, if the lesser fitness value is obtained based on particle position, pb and pbest are updated. Here, in this case time,t represent the iteration or generation. New particle velocity and position at time t+1, is calculated using Equation (1) and Equation (2). As the iterations proceeds, until the fitness value of the particles converges or exit condition is met.

4.2 Instrumentation of Code

The code is transformed by eliminating all the unnecessary body of the code. The entire body of the code that comes after the target node is eliminated. All the necessary elements for calculating the fitness value are included. Branch distance and approach level are calculated. Branch distance is calculated using the Korel's formula. It is

```
import java.util.Scanner;
public class Sourcetri {
public static void main(String args[]){
int a,b,c;
Scanner o=new Scanner(System.in);
System.out.println("Ener three sides of triangle");
a=o.nextInt();
b=o.nextInt();
c=o.nextInt();
if(a>0 && b>0 && c>0){          //node1
          if(2*a<a+b+c ||2*b<a+b+c || 2*c<a+b+c){       //node2
             if(a!=b) {          //node3
                    if(a!=c){  //node4
                       if(b!=c)         //node 5
                           System.out.println("\n b!=c Scalene");
                         else if(b==c) //node6
                         System.out.println("\n b==c Isoscelas");
                       }
                       else if(a==c)   //node7
                         System.out.println("\n a==c Isoscelas");
                 }
          else if(a==b){         //node8
                  if(b!=c)       //node9
                         System.out.println("\n b!=c Isoscelas");
          else if(b==c) //node 10
                         System.out.println("\n b==c equilateral");
                  }
        }
}
else    //node 11
     System.out.println("\n no triangle");
  }
}
```

Fig:6 Instrumented Code for SUT1

calculated just before each if predicate. Number of if predicates that needs to be encountered before reaching the target node is the approach level. The approach level for target node is set as zero. Finally the fitness value is calculated based on branch distance and approach level. Instrumented code is implemented in fitnescal() function. Instrumented code for triangle classification program is shown in Figure 6.

4.3 Implementation of PSO

Initialization of particles Particles are initialized using function par-ticleinit(). The test data values are generated by using random num-ber generator within the given limit. Velocity values corresponding to each test data are also initialized using random number genera-tor. PSO algorithm Once the particles are initialized the fitness is calculated using the fitnescal() fuction. Then the test data values are updated using the velocity update equation in PSO. pso()function per-forms the repeated calculations of pbest and gbest values in iterations.

4.4 SUT 1: Triangle Classification Program

The program which takes 3 sides as input and determine whether the triangle is isosceles, scalene or equilateral is considered as program to be tested. There are six nodes which can be set as target nodes. These include one for scalene, three for isosceles, one for equilateral and one for no Triangle. One node is chosen as target node and PSO algorithm is applied to produce the best test data. The program takes 10 particles initially and applies PSO in 10 epochs. Each particle consists of a, b and c which are the 3 sides

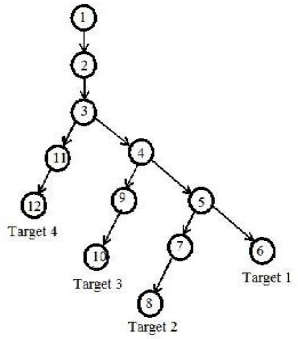


Fig:7 CFG fot SUT1 of triangle.

The control flow graph for SUT 1 is represented in Figure 7.

4.5 SUT 2: QUADRATIC EQUATION SOLVER

This SUT classifies the given input data as roots are imaginary, root is real and equal, roots are real and unequal. Control flow graph of this SUT has 6 Nodes with 3 predicate nodes with maximum nesting level of two.

4.6 SUT 3: Sthamer's Triangle Classifier Problem

This SUT characterizes a triangle based on its info sides as isosceles, equilateral, right edge triangle or scalene. Three sides of the triangle are given as contribution to this SUT. Control stream chart of this SUT has 28 Nodes with 13 predicate hubs with most extreme settling level of eight. It has correspondence conditions with AND administrator and complex social administrators.

4.7 Parameter Tuning in Particle Swarm

A few parameters associated with PSO can be refined so the execution of the calculation is progressed. These incorporate restricting the greatest speed, choosing quickening constants, the choking factor and the idleness consistent.

4.7.1 Selection of Maximum Velocity

Toward the finish of every emphasis speed of every molecule is refreshed utilizing refresh condition which relies upon individual best and worldwide best of every molecule. In the event that the speed isn't controlled, it might make the particles take after more extensive cycles in the hunt space. To keep this uncontrolled development, upper and lower points of confinement can be connected to speed of every molecule once the refresh condition delivers new speed. Where xmax and xmin are greatest and least esteems discovered so far by every molecule n is normally taken as 2.

4.7.2 Selection of Acceleration Constants

Accelerating constants control the improvement of each atom towards its individual and overall best position, independently.

Little regards confine the advancement of the particles, while broad numbers may influence the particles to veer. Ozcan and Mohan drove a couple of examinations for the exceptional example of a single atom in a one-dimensional issue space to investigate the effect of a deterministic expanding speed relentless. The makers deduced that by a development in the estimation of the enlivening relentless, the repeat of the movements around the perfect point increases. For tinier estimations of constants, the case of the course resembles a sinusoidal waveform; in any case, if the regard is extended, the unpredictable methods for interlaced cyclic bearings appear. The bearing goes to unendingness for estimations of more critical than 4.0. Decision of fixing segment or inactivity enduring Empirical examinations performed on PSO exhibit that despite when the best speed and expanding speed constants are successfully portrayed, the particles may regardless meander, i.e., go to boundlessness; a marvels known as blast of the swarm.

4.7.3 Constriction Factor

The main technique to control the blast of the swarm was created by Clercs and Kennedy. It presents a narrowing coefficient which in the least difficult case is called Type 1. All in all, when a few particles are considered in a multidimensional issue space, Clercs strategy prompts the accompanying refresh run the show.

4.7.4 Inertia Weight

Second strategy proposed by Shi and Eberhart recommends another parameter which will just duplicate The speed at past time step.ie vi(t-1). This parameter is deciphered as dormancy steady(ic) Suitable determination of idleness consistent outcomes in brisk joining. Extensive esteems encourages worldwide investigation though little weight factor encourages nearby investigation. Along these lines substantial weight factor is decided for introductory age and steadily littler weight factor is decided for progressive ages. Following capacity is utilized for dynamic dormancy weight. Diagram in Figure 5.3, plots wellness esteems against number of emphasess. It can be watched that the wellness esteem vacillates with dif

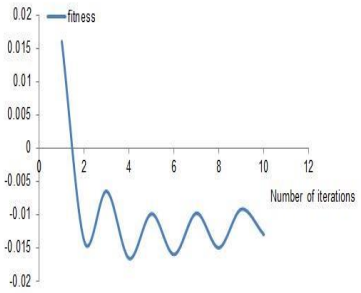


Fig: 8 Fitness Value vs no. Iterations

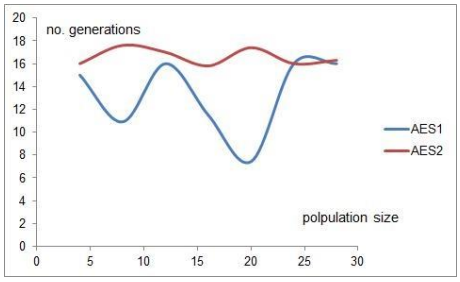


Fig: 9 Population Size vs no. Iterations for Conver

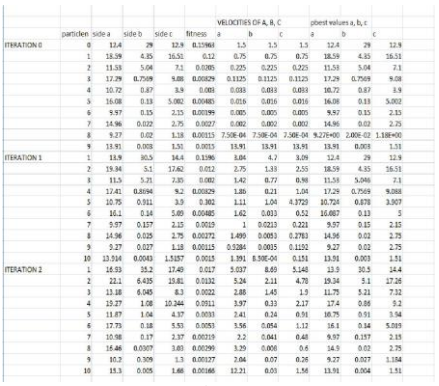


Fig: 10 Result for Iterations 0 to 2

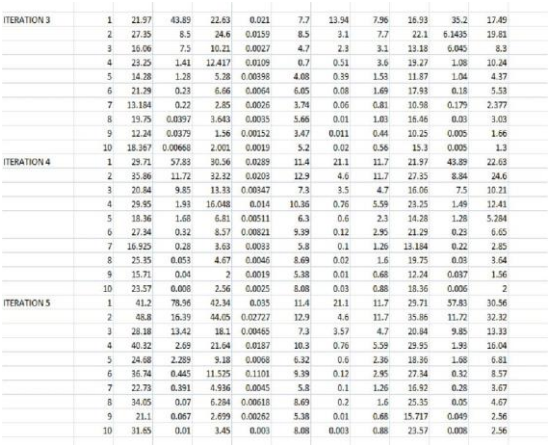


Fig:11 Result for Iterations 3 to 5

different cycle numbers. Populace estimate was fluctuated and meeting was checked for 2 SUTs. Chart in Figure 5.4, plots populace measure against number of emphasess required for merging for 2 distinctive SUTS. It can be watched that the variety is arbitrary. Subsequently it can be inferred that there is no huge impact of populace measure on union. The test information created for SUT 1 for 10 emphasess is demonstrated as follows. Result for emphasis 0 to 2 is spoken to in Figure 5.5. Result for cycle 3 to 5 is spoken to in Figure 11.

5. Conclusions

For a given code a set of test data which satisfies the given condition is generated and optimal data is found within the search space. The data with least value of fitness function is termed as optimal data. The algorithm was fine tuned by applying parameter tuning and the results were studied. It was observed that the algorithm with tuned parameters outperformed the conventional algorithm. It was applied to various software under tests and many trials were conducted. The population size was varied and the convergence of the algorithm was studied. It was observed that population size does not have a significant effect on convergence. Irrespective of population size PSO performs effectively. Success rate of the algorithm was checked and it was found to be high. Thus it can be concluded that PSO is efficient for test data generation.

References

[1] M. Harman,P. Mcminn, "A Theoretical and Empirical Study of Search-Based Testing:Local, Global, and Hybrid Search", IEEE Transactions on Software Engineering, 36(2), 2010, pp. 226-247.

[2] M. Xiao, M. El-Attar, "Empirical evaluation of optimization algorithms when used in goal-oriented automated test data generation techniques", Springer Science + Business Media, LLC 2006.

[3] M. Harman, K. Lakhotia, P. McMinn, A Multi-Objective Approach to Search-Based Test Data Generation , Proceeding Genetic and Evolutionary Computation Conference,2007, pp. 1098-1105.
[4] B. Korel, ”Automated Software Test Data Generation”, IEEE Transaction Software Engineering,16(2), 1990, pp 870-879.
[5] P. McMinn, ”Search-Based Software Test Data Generation: A Survey”, Software Testing, Verification and Reliability, 14(2), 2004, pp. 105-156.
[6] V. Yamille,K Ganesh, M. Salman,”Particle Swarm Optimization: Basic Concepts, Variants and Applications in Power Systems”, IEEE Transactions on Evolutionary Computation,12(2), 2008,
[7] H Xiohui, E. Ressel, ”Multiobjective optimization using dynamic neighbourhood”, Department of Biomedical Engineering, Department of Electrical and Computer Engineering, Purdue University, Indiana, USA