iJRASET
International Journal For Research in Applied Science and Engineering Technology

INTERNATIONAL JOURNAL FOR RESEARCH

IN APPLIED SCIENCE & ENGINEERING TECHNOLOGY

Volume: 6 **Issue:** VII **Month of publication:** July 2018

DOI: http://doi.org/10.22214/ijraset.2018.7017

www.ijraset.com

Call: 08813907089 | E-mail ID: ijraset@gmail.com

International Journal for Research in Applied Science & Engineering Technology (IJRASET)
ISSN: 2321-9653; IC Value: 45.98; SJ Impact Factor: 6.887
Volume 6 Issue VII, July 2018- Available at www.ijraset.com

Automatic Test Case Generation by using Parallel 3 Parent Genetic Algorithm

Nargis Akhter[1], Dr. Amar Singh[2], Mr. Guljar Singh[3]
[1]*Research Scholar,* [2,3]*Baddi University of Emerging Science and Technology, Baddi, India*

Abstract: ***The software testing is one of the most predominant and major phase of software development life cycle ensuring the verification and validation process of the software. Test cases should have capability to cover more test objective features i.e., achieve more test coverage. The distinct method for automatic generation of test cases find trivial set of test cases decrease labors as well as budget of software testing and produce maximum effective and operating testing of software product. The most critical step in software development process is the generation of test data. Many methods have been established by researchers to automate it. In this paper we investigate that how P3PGA algorithm improves the quality of the test data and make it more efficient to cover the maximum paths***
Keywords: Test Data, Test Case, GA, P3PGA

I. INTRODUCTION

Software testing is an authoritative and self-motivated part of the software development process [1]. Software testing is very man power consuming and expensive process [2]. The testing efforts are classified in to several parts: test case generation, test execution, test evaluation [1]. Test case generation is the fundamental part of any testing process and automating it save copious time effort as well as losses the no of incorrectness and mistakes.

An exactly created test suit may not only uncover the faults in a software system but also helps in dropping the high cost, efforts associate with software testing [3]. Software testing plays an important role in high quality software development. It practices the application of artificial intelligence techniques.

Soft computing is an emerging methodology to computing, whose purpose is to exploit easiness for inaccuracy, indistinctness and partial truth to achieve robustness, submissiveness and total low cost [1]. Soft computing are the techniques with in computer science to find out the results to computationally –hard tasks such as the solution of NP-hard problems for which a precise solution should not be derived in polynomial time.

There be present a number of techniques for producing test cases like fuzzy logic, finite state machine, soft computing, genetic programing and evolutionary computation [4] . There are two techniques which are used for automated test data generation such as functional and Structural testing. Structural testing are further divided into sub categories which includes different kind of approaches and implementation methods such as Random approach, Path-oriented approach, Goal-oriented approach, Static method, Dynamic method, Hybrid method, Intelligent Approach [5].There are some techniques of test case generation that be determined by on the application like test case generation for web application, object oriented application, structured based system, UML applications etc.

1) *UML diagrams*: from system requirements we can derive test cases and we can also derive test cases from use cases [6]. Some use case models are used for production of test cases. Use case Diagram Graph (UDG) is a diagram for originating test case that demonstrates all the possible case in the SUT.
2) *Critical Path Method*: T.Y. Chen et al [7] projected a framework for producing test cases i.e. critical path method. By using this technique we generate test cases by purifying functional choices.
3) *Code based test Generation*: [A. Mathur] test cases can be produced directly from the code which is called as code based test generation. This method decreases the size of the test suite, or prioritizes tests by supporting regression testing.
4) *Dynamic Path Testing and Evolutionary Technique*: it is dynamic path testing technique that engenders test case by implementing the program with different probable test case values. Korel also offered a methodology of test case prioritization established on path testing. With the help of random testing test cases are organized according to the smallest path [8]. J. Wegener and H.Sthamer generate test cases with the help of several structural test coverage testing criteria by evolutionary approaches like Genetic algorithm, which frequently generate test cases by using Meta heuristic method.

ISSN: 2321-9653; IC Value: 45.98; SJ Impact Factor: 6.887
Volume 6 Issue VII, July 2018- Available at www.ijraset.com

II. GENETIC ALGORITHM FOR TEST DATA GENERATION

The most current exploration is being done on test case generation using Genetic Algorithm. An optimization technique like GA can be used to solve various problems. It uses existence of the fittest technique, where the best solution survives. Genetic Algorithm is based on "SURVIVAL OF FITTEST". In genetic algorithm we represent input by set of chromosomes. These chromosomes are represented in different numbers in computer. The mostly used representation of chromosomes is binary representation. Basic steps of genetic algorithm are shown in fig2.1 [9]:-

A. [Start] Generate random population or n chromosomes.
B. [Fitness] Evaluate the fitness f(x) of each chromosome x in the population.
C. [New population] Create a new population by repeating following steps until the new population is complete.
1) [Selection] select two parent chromosomes from a population according to their fitness
2) [Crossover] with a crossover probability cross over the parents to form new offspring. If no crossover was performed, offspring is the exact copy of parents.
3) [Mutation] with a mutation probability mutate new offspring at each locus.
D. [Accepting] Place new offspring in the new population.
E. [Replace] Use new generated population for a further run of the algorithm.
F. [Test] if the end condition is satisfied, stop, and return the best solution in current population.

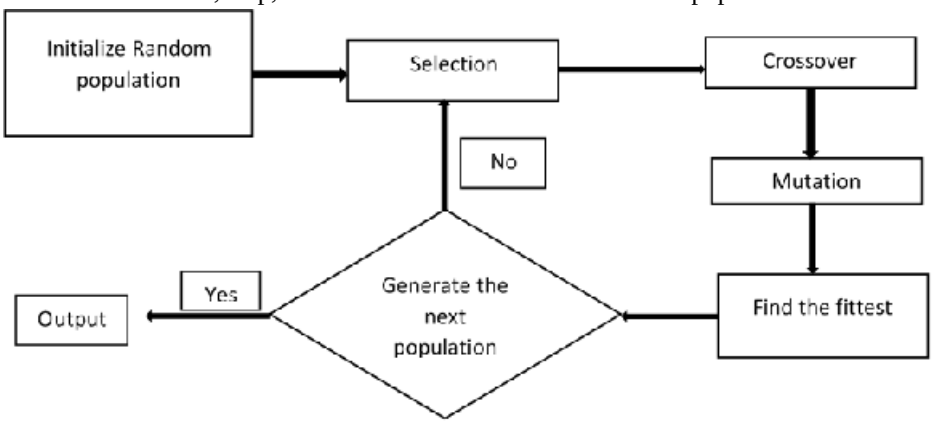


Fig. 1 Flow chart for basics steps of ga

III. PARALLEL 3 PARENT GENETIC ALGORITHM

P3PGA algorithm is a multi-population algorithm in which evolution process takes place on many populations in parallel [10]. It is based upon the single three parent genetic algorithm.

Start
Generate n populations randomly by chromosomes
For Gen =1: Number of Generations
For i = 1: N
Make mitochondrial change to $i^{th\ population}$ to generate 3Parent (3-P) population.
//* we implement this process by adding a small random number in every gene of the individuals.*//
Add current ith two parent (2-P) population with new 3-P population.
Calculate fitness, sort population and select best fittest individuals.
Find and Record fittest solution.
Produce new ith 2-p population by applying general Genetic Algorithm:-

1) Initialize the population randomly by chromosomes.
2) Evaluate the Fitness and compute elite.
3) If satisfies the fitness function than stop otherwise go to next step.
4) Generate the new population around the elite by adding or subtracting a small number in it.
5) Repeat step four.

Check bounds violation & correct if needed.
Select local best candidates $l_{best(i)}$ for i^{th} population;

End for
From amongst the local N best candidates select the globally best g_{best} candidate;
For I =1: N do //* move local best towards global best
With a given probability replace a gene of $l_{best(i)}$ with the corresponding gene of global best ($g_{best(i)}$) candidate;
End for
End for
End

IV. TEST DATA GENERATION USING P3PGA

The aim of this research is to propose a method for generating test data automatically to achieve path coverage that guarantees coverage of all paths of the program under test by using P3PGA. Basically this approach is an enhancement of Genetic Algorithm. In this algorithm we can take N number of populations rather than a single population to generate test data. Flow graph of P3PGA is shown below in Figure 4.1
Initialize the random population N.
After Generating the N population perform the mitochondrial changes on the ith population. Mitochondrial changes are small changes which are performed by adding a small number in the population. After the mitochondrial changes a new population is generated which is 3P population. Now combine the ith population with 3P Population. And find out the fittest among them. Fitness of each test data for N population based upon paths which are covered by that test data. Now Record the set of fittest solutions. Now we have a population set of fittest values.
Now to generate new ith 2-P population select the fit individuals for recombining. Now perform the crossover on the population. After crossover we have a new population. Combine this population with the fittest set of population. The population become double. Select the best fit test data out of them. We have N number of local best tests data. From these local best tests data select the global best test data. The global best test data is that local best test data which has greater fitness value. After half of iterations we can move local best test data towards global best test data. For this purpose we can generate a random number between 0 and 1. If the random number which is generated is greater than 0.5 than that correspondence gene of local best data interchange with correspondence global best gene.
After all iteration we have set of fittest test data for maximum path coverage.

V. ALGORITHM FOR P3PGA TEST DATA GENERATION:

Start
Initialize the random population N for multiple test data
For Gen= 1: Number of Gen
For i= 1: N
Generate the random number
//*Now make the mitochondrial changes in ith population to produce 3P populations*//
Combine initial population with the population after mitochondrial changes.
Evaluate the fitness
Sort the population's sets
And select the best fit test data
Now generate the new populations
Select fit individuals for recombining
Now perform the crossover on this population
//*We have set of population which is double of initial populations*//
Evaluate the fitness
Select the $global_{best}$ test data from $local_{best}$ test data
If the iterations [k >= k/2] then
Move towards $global_{best}$ test data
Generate random number between 0 and 1
If the no is >0.5 then
End

ISSN: 2321-9653; IC Value: 45.98; SJ Impact Factor: 6.887
Volume 6 Issue VII, July 2018- Available at www.ijraset.com

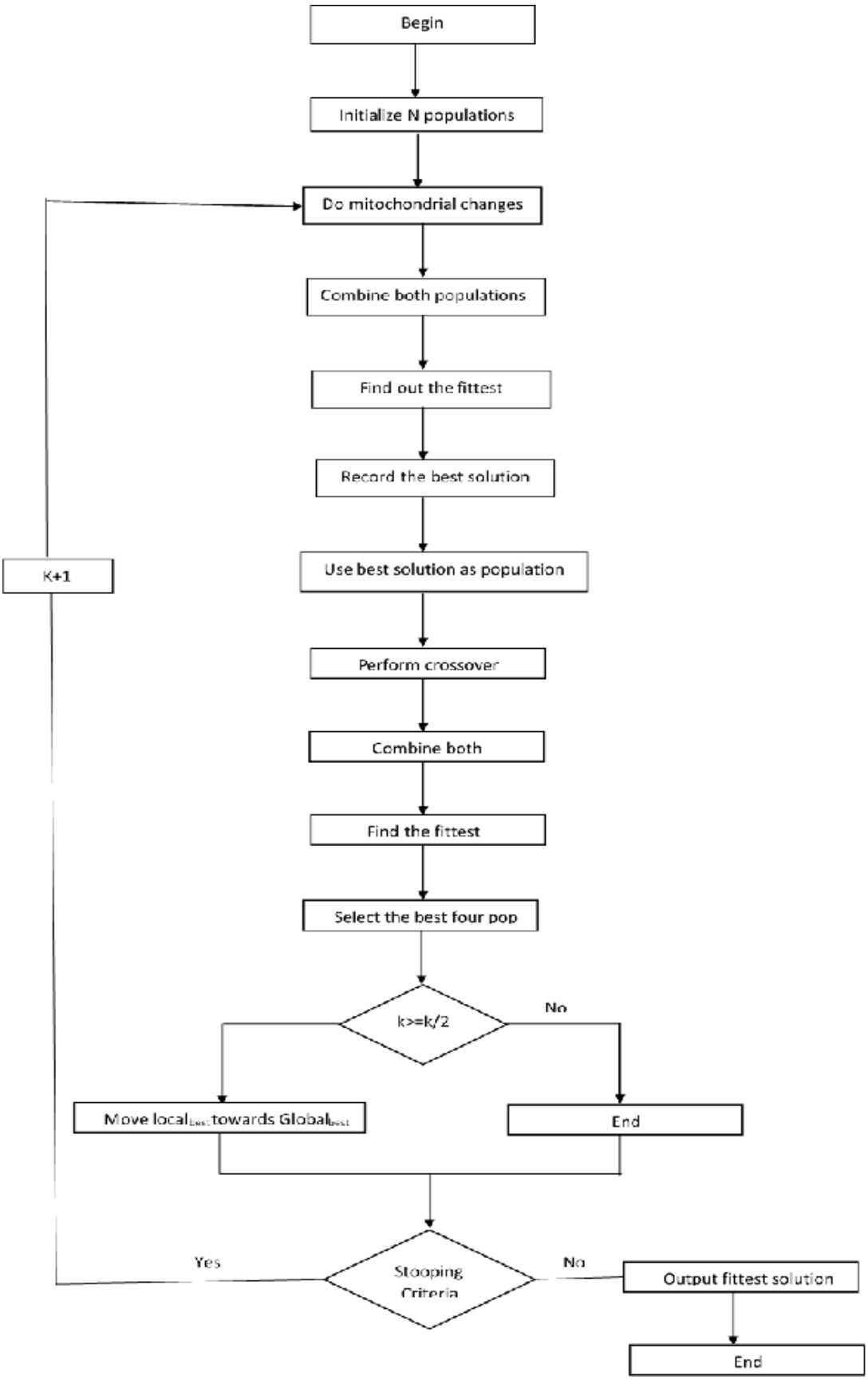


Fig.2 CFG of Test Data Generation by P3PGA

ISSN: 2321-9653; IC Value: 45.98; SJ Impact Factor: 6.887
Volume 6 Issue VII, July 2018- Available at www.ijraset.com

VI. RESULTS AND SIMULATIONS

In our work GA and Parallel GA are used to generate test data. In order to investigate the performance of both GA and parallel GA we implement these algorithms on real world problems. The CFG of the program are constructed manually from respective source code and all feasible paths are identified manually. . We implemented the Genetic Algorithm and Multi Population Genetic (P3PGA) Algorithm in MATLAB and tested its performance on various program of MATLAB [11].The program which we are using in our work are given below:-

Prime Number (Isprime): - In this program we generate test data to check weather a number is prime or not. Or not if yes then identify the type of triangle.

Quadratic equation (quad): - In quadratic equation program we check whether the given input is of a quadratic equation or not if yes then find out the roots. This program also finds out that the equation is linear or infeasible.

Car Brake Controller system (CBCS):- we can use this program to identify there is any requirement of brake to stop the car or not by using speed and time, if yes then which type of brake is required at that time to stop car at a certain point. CFG of all programs are given below: - (fig 4.1 to 4.4)

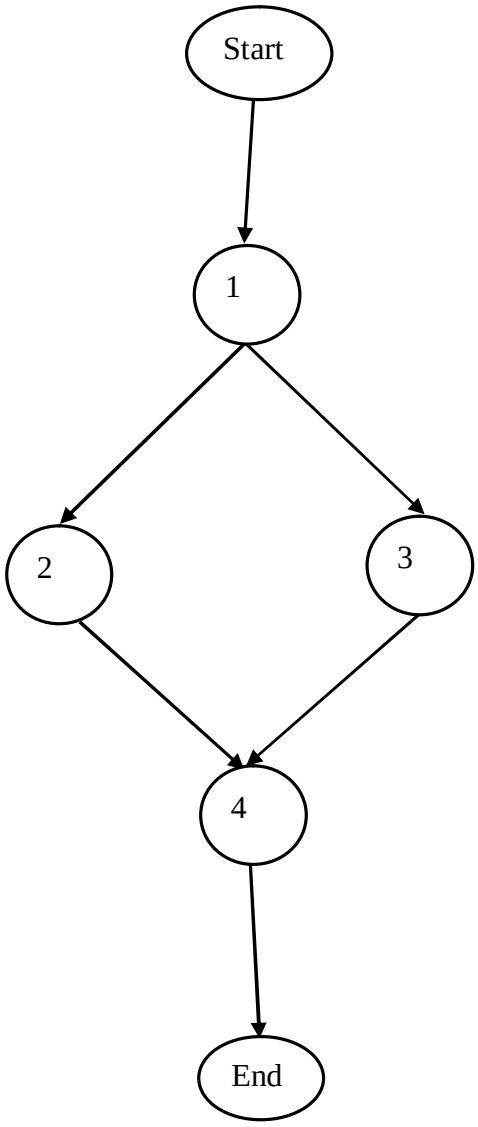


Fig. 2 CFG of Isprime

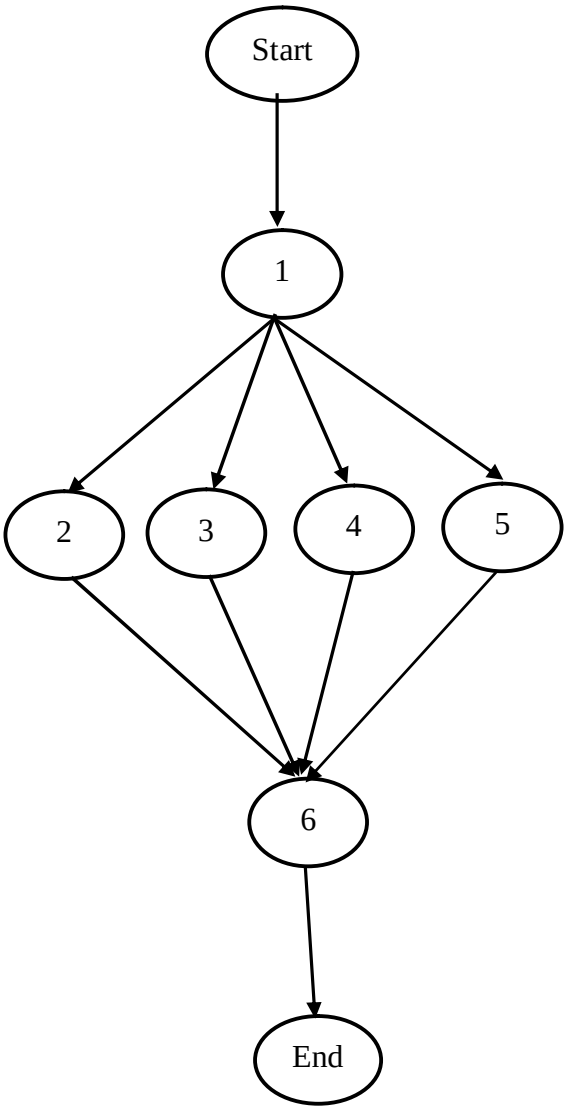


Fig. 3 CFG of Triclsfr

ISSN: 2321-9653; IC Value: 45.98; SJ Impact Factor: 6.887
Volume 6 Issue VII, July 2018- Available at www.ijraset.com

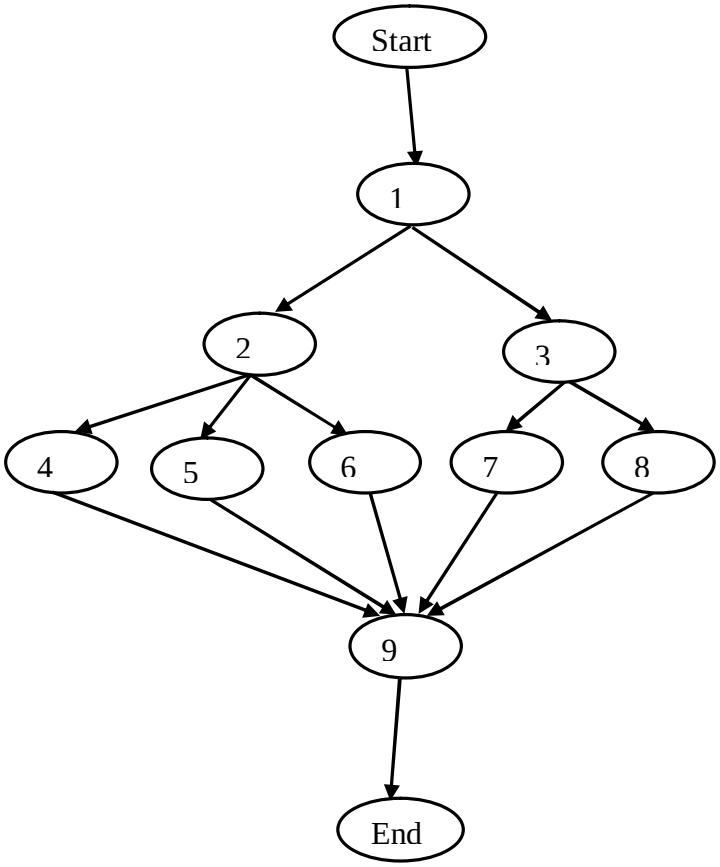


Fig. 4 CFG of QUAD

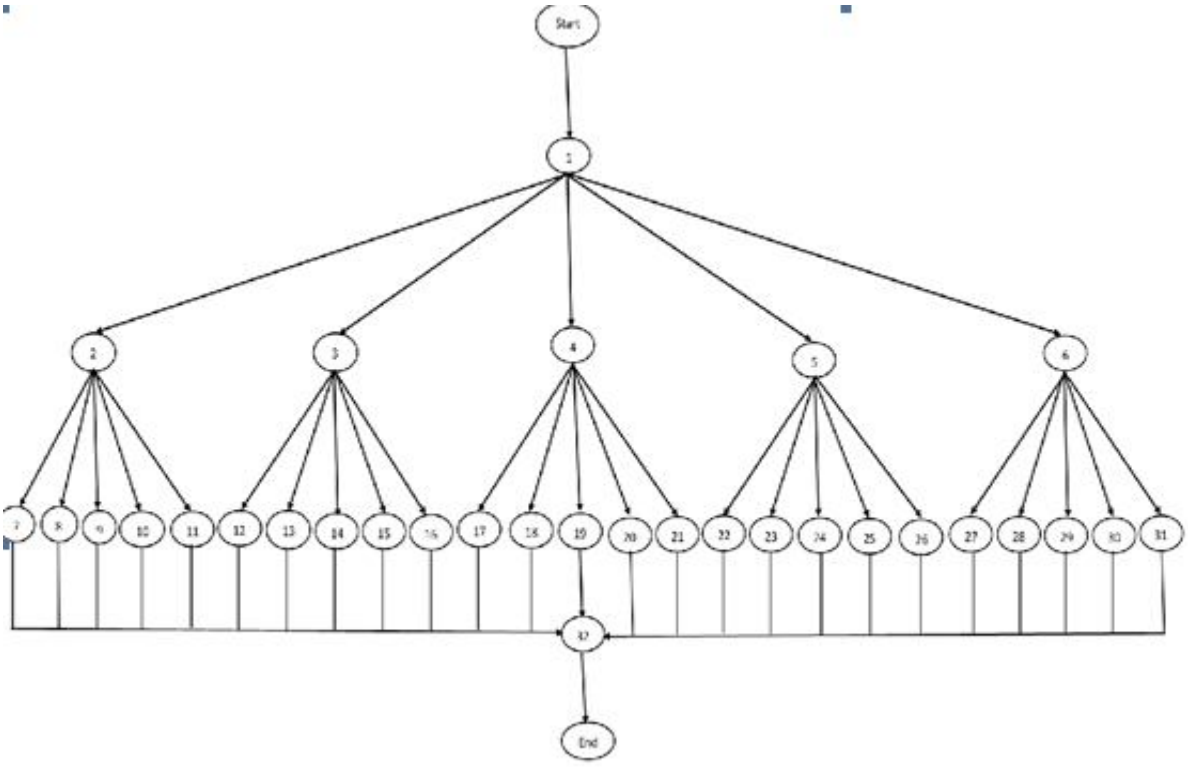


Fig. 5 CFG of CBCS

For each program we have conducted 20 trials the average percentage coverage of each program shown in table 4.1

Table 1 Average %age coverage by both GA and P3PGA

Program Name	Average %age coverage by Genetic Algorithm	Average % age Coverage by P3PGA
Prime Number	75%	100%
Triangle Classifier	81.25%	100%
Quadratic Equation	76%	100%
Car Break Controller system	83.25%	96.8%

Table 4.1 shows that the Average percentage coverage of Prime Number by Genetic Algorithm is 75% and the Average percentage Coverage of Prime Number by P3PGA is 100%. Average Percentage Coverage of Triangle Classifier by GA is 81.25% and the Average %age Coverage of Triangle Classifier by P3PGA is 100%. The Average Percentage Coverage of Quadratic Equation by GA is 76% and the Average %age Coverage of Quadratic Equation by P3PGA is 100%. Average Percentage Coverage of CBCS by GA is 83.25% and the Average %age Coverage of CBCS by P3PGA is 96.8%

Experimental results of each program by both approaches shown in figure 4.5 in the form of chart:-

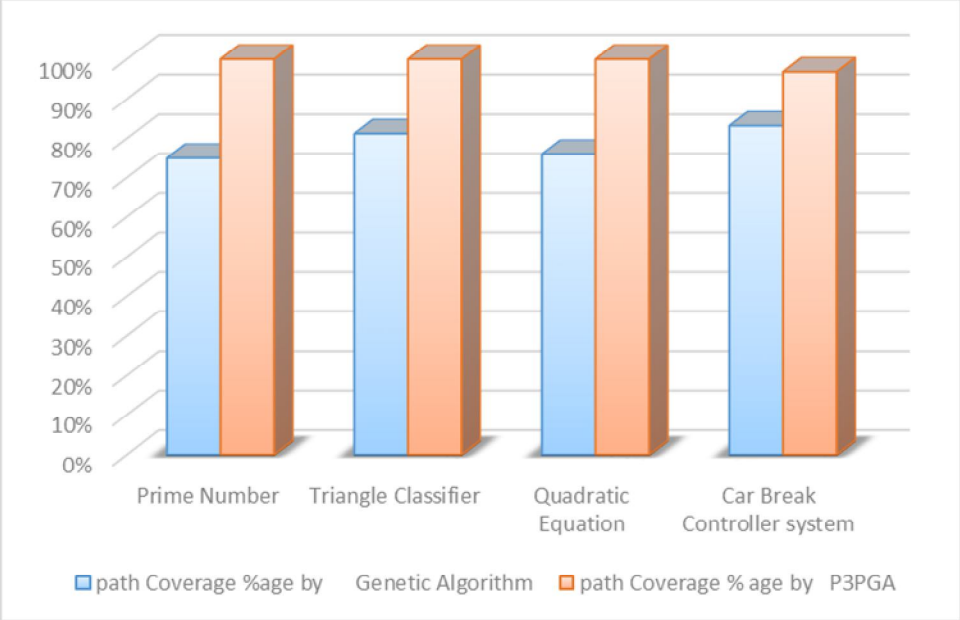


Fig 6 average %age coverage of all programs

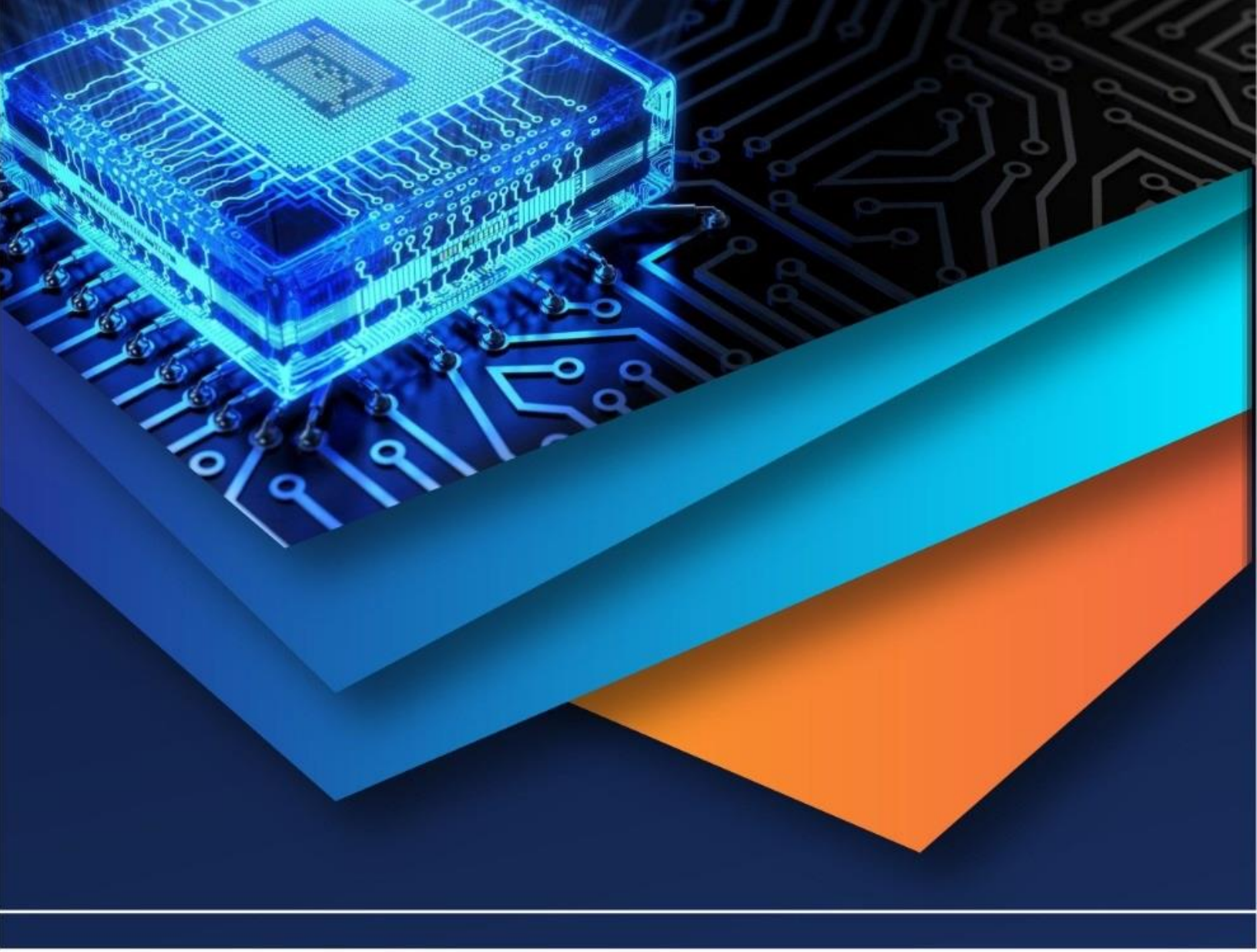
VII. CONCLUSION

In this paper we proposed an evaluated test data generation technique that work on principle of Genetic algorithm for test data generation that provide better performance and results than GA. PB3GA is an advanced form of GA for more efficient test data Generation automatically. PB3GA is multiple population algorithm and highly suitable for complex problems. The simulation results clearly shows that PB3GA approach have better performance than GA base approach.

REFERENCES

[1] Pakinam N. Boghdady, Nagwa L. Badr, Mohamed Hashem and Mohamed F.Tolba. A Proposed Test Case Generation Technique Based on Activity Diagrams. International Journal of Engineering and Technology.2011
[2] Manoj Kumar, Arun Sharma and Rajesh Kumar. Optimization of Test Cases using Soft Computing Technique: A Critical Review. Issue11, Volume 8 WSEAS Transaction on Information Sciences and Applications. 2011
[3] Priyanka Bansal. A Critical Review on Test Case Prioritization and Optimization using Soft Computing Techniques. ICRTNB 2013.

[4] Itti Hooda and Rajender Chillar. A Review: Study of Test Case Generation techniques. International Journal of Computer Applications 2014.
[5] Naveen Shaillu and Dr. Amar Singh. Soft Computing Based Approaches for Test Data Generation, A survey. International Journal of Creative research and Thoughts. Volume 6, Issue 2 April 2018.
[6] Philip Samuel, Rajib Mall, A Novel Test Case Design Technique Using Dynamic Slicing of UML Sequence Diagrams, e-Informatics: Software Engineering Journal, Vol 2, Issue 1, 2008.
[7] J, C. Huang. An Approach to Program Testing. 1975.
[8] Korel.B, L.Tahat and M.Harman, Test Prioritization using System Models, in proceedings of the 21st IEEE International Conference on Software Maintenance (ICSM 2005), page 559-568, September 2005.
[9] K.Singh, R.Kumar, "Optimization of Functional Testing using Genetic Algorithms" International Journal of Innovation, Management and Technology, vol. 1,no. 1, pp. 43-46, ISSN: 2010-0248, Apr. 2010.
[10] Amar Singh, Sukhbir Singh Walia, Shakti Kumar. P3PGA: Multi-Population 3 Parent Genetic Algorithm and its Application to Routing in WMNs. International General of Advance Result in Computer Science. Volume 8, No. 5, May- June 2017
[11] Shayma Mustafa, Radziah Mohamad, Automatic Test Case Generation for Structural Testing Using Negative Selection Algorithm. IRICT 2014.



10.22214/IJRASET

45.98

IMPACT FACTOR: 7.129

IMPACT FACTOR: 7.429

INTERNATIONAL JOURNAL FOR RESEARCH

IN APPLIED SCIENCE & ENGINEERING TECHNOLOGY