Search based software testing with genetic using fitness function

Mr.Gautam M Lodha
ME(Research scholar)
Dept.of computer science and engineering
G. f 'college of engineering
Jalgaon,india
gml786@rediffmail.com

Mr.Rahul S Gaikwad
Asst. Prof. Information Technology
Dept.of Information technology
G. f' college of engineering
Jalgaon,india
Gaikwad005@gmail.com

Abstract— **software engineering is deal with development activity in which testing is an important part .in testing if it is possible to generate an test case for code then it is easy task to find error in program so white box testing is possible using search base algorithm.in software testing white box testing is test an code which is used to develop an software. In this article purposed system is use genetic algorithm.Software testing is branch of software engineering in which testing performing vital role in any kind of software system .basically testing is noting but to find out bug as early as possible. in other words testing is intention to find out the error from software program. Software testing is an essential but expensive activity in software development life cycle and hence much research effort has been put into automation of software testing .In software testing it is interested to see how well a series of test input test a piece of code with intention to find out bug.**

Keywords— Ranodm serach,Hill climbing,Genetic algorithm fitness function mutation,cross over.

I. INTRODUCTION

In software engineering it is necessary to carried out testing in each and every phase of system development of life cycle. Because without software testing the system development life cycle can not be completed. An error or defect in software or hardware can create serious situation in running any software program that error can scientifically known as bug which occur in source program there are various method by which reduce the error or bug these can be improvement in programming style ,programming techniques development methodologies and some kind of language support. In software testing there are various methodology involve to test source program. These methodology are discuss as follows .

1)Black Box Testing. 2)White box testing.

3)Unit testing. 4)Integration testing.

Black box testing **:** Black box is treat system as literal Black-box.so it doesn't explicitly use knowledge of internal code and source program. it is usually described as for as testing functional requirement. Black box include behavioral, functional ,and close box testing. Black box testing is also called functional testing. Define initial component state, input and expected output for the test.

Following are some advantage of Black box testing they are as follows.

1)Easy testing In case of black box testing it is easy to test software externally because no need of developer or expert is required to test an source program. also no need of source program knowledge, only User interface is required to test in Black box testing.

2)Quick Result In case of black box testing it is very important that because only testing of graphical user interface is involve and hence quick result is possible.

White Box Testing: Testing is all about telling people about the quality of the system under test .it was necessary to focus on test case generated on code and finally produce branch coverage. Also tell people about bug inside the code and behavior of the system to ensure that project knowledge that what the purposed system can do and allow stakeholder to take decision about improvement about system quality. allow one to peak inside the box and it focuses specifically on using internal knowledge of the software to guide the selection of test data .following are different coverage method of white box testing.

Code coverage: has each line of source code been executed.

Function coverage: has each function in program can be executed.

Debugging: will always be white-box testing Loops are the great problem.

Integration testing : Integration testing involvement of working of module and incorporate them. Hence module must be tied together to form particular system. how there working is carried out and what information is interface so that they use to develop a system. Also it need to make sure that the interface contain correct content and provide perfect testing.

Unit testing: while if in case of unit testing it is necessary to run particular module correctly and make sure that test were structured and potentially recorded correctly for future use.
In case of software testing there are various tools are available for testing software but these software tool are work on black box testing like QTP, load runner .also here are some software tool are work on white box testing like Evosuite, Jtest but they have some less comfortable. In this article we are going explain such kind of system that would provide automatically generation of test case and on the basis of test case provide best test case to generate optimal solution for source program input.

II. RELETED WORK

N.willams B.marre[1]provide "on the fly generation of k-paths test for c condition .give an certain information about gray box testing and black box testing with there various method.
P.McMinn[2] introduce an anlysis of "test data generation" in behavioral structure in which they provide test data for set of test case. also provide the work on automated test data and verification and validation testing with reliability of testing.
A.Leitner M Oriol[3] provide an information related to unit testing in which give us an information how to execute an unit or module and to test that module also how efficiently an test case minimization is useful for automation test tool.
J.H.Andrews,A.groce[3] has provide an information about "random test run length and effectiveness" and there various method with there implementation.
A. Arcuri and X. Yao[4] "Search Based Software Testing of Object- Oriented Containers," provide data related to object containers data and search based testing. Also Arcuri provide an full runtime analysis variable method on the triangle classification problem" .
Richard torkar and Wasif Afzal[5] are work on non functional properties and provide detail regarding to Execution time,Qos and usability testing which help us for our future work.
MR. Bares[7] and M Miraz[7] has work on evlutionary algorithm hence easy to work on genetic programming. In which working of evolutionary system provide only non functional properties such as black box testing .but they are not worked on length of test case and measurement of code.
Marinow and D. Notkin[8] provide much more information about unit test in which he carried out the work on unit testing and carried out work on unit or module wise testing.
S.kurshid[9] give us java path finder and theory of complex data structure and give information about black box testing. Also provide how to use java path finder for search based algorithm and give us fruit fool information . in the area of search based software testing.
Yury pavlu[10]and Gordon Fraser[10] give information about semi automatic search based test generation .in that he provide information about how useful is search based and efficiently generate test case for high code coverage.
Andrea Arcuri "A Theoretical and Emprical[11] analysis of Role of Test Sequence Length In software testing for Structural coverage"give an information related to test case.

III. PROPOSED WORK

A Algorithm.
In search based software testing we are going to perform work on test case generation. In which we fully concentrate on white box testing. which is useful for the software developer and software tester to test the software. Hence mainly we focus on structural code coverage and branch coverage. In this article we first use genetic algorithm with mutation and fitness function to calculate population of test case. The population is nothing but test case also known as test gene which provide deterministic result for optimal solution. We can use rank based selection with valid value such that high chance to select them for testing. here we provide rank to test sequence and from that test sequence or test case we can search particular test data generation of test input.

B Equations and formula's

The equations use to generate a test case and to calculate an mutation are as follows

1)Mutation can be calculated with following formula
a=b(c)+1
where a=test variable, b=test case data, c=test case no.1=constant value to mutate with no.
2)Test Case Generation:
Test case can be genrated with help of random function i.e.as follows
Random r = new Random();

C Algorithm & code

```
for (int tstCnt = 0; tstCnt < 20;
tstCnt++)              {
generatePopulation();
btnExecuteTest_Click(sender, e);
btnExecute_Click(sender, e);
button1_Click_1(sender, e);}
readBranchCoverage();
calculateFittestGene();
MessageBox.Show("Fittest Test
Case No : " + (fittestTestCase +
1) + "Test Case :" +
testCaseData[fittestTestCase + 1,
1] + "," +
testCaseData[fittestTestCase + 1,
2]);
for (int j = 0; j <
testCaseVarCount; j++)
```

Fig No: 1

D Contributed work

In purposed system we are going to design an system that will generate an automatically test case .we use fitness function and mutation concept of genetic real time system in which we provide best optimal test case that would cover total source code and provide maximum branch coverage and structural coverage for the source program.

E Abbreviations and Acronyms

GA :Genetic algorithm

FF: Fitness function

EA: Evolution approach

SBST: Search Based Software Testing

MT: Mutation.

HA: Heuristic approach.

F Algorithmic step :

In this section I am going to explain algorithmic step for my candidate system As follows

Step 1:Give the input to system

Give the source input to my system the source input is c source file that code written in c/cpp file. the written code in c programming must be correct if the written code is incorrect then test case cant generated by purposed system.

Step2:Genrate test case using random search algorithm

Random search algorithm is used to generate test case given source input c program can be spited in term of token after token is generated these token can be identified by and variable can be identified by candidate system. these candidate system can generate test cases using random function that means randomly integer value can be generated for ex. Variable value range 1-999 .After generating test case using random function. These test cases known as test gene from which we can create population having group of no of test case.

Step 3 Creating population of test case as test gene

Using test case we can create an test gene these test gene are known as population that can be created using random function which explain earlier.

Step 4 fitness function to find best test case from test case.

In this step I am going to use concept of genetic algorithm with fitness function . in fitness function of GA the best test case can be selected to find maximum branch coverage in source program.

Step 5 concept of Mutation and Evolution approach to find

Using mutation and evolutionary approach of purposed system it is possible to find best tests[2,3] cases among fittest value provided by fitness function.

For ex. If in above example the test case as follows

Fittest value test case 1	93	27	95
Mutation value test case 1	94	28	96
Mutation value test case 2	95	29	97

These mutation value can be calculated with mutant integer 2.

Step 6 Exit

G Flow of Purposed System

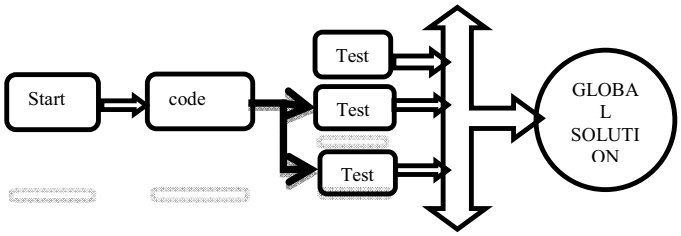

Fig no. 2

In the purposed system our input is user source code which present in c/cpp file. In this article we first taking an control over source program through our purposed system to generating various test case as purpose system stated. With that we can generate various test case known as population. We can match expected result with theoretical result. if system found correct result then correct report is generated. but if any error or bug is found in the user source code then purposed system can't accept source code. Above figure show the actual flow of purpose system in which first we have source program available in c cpp as input that will be given to our system which produce test case that test case will be save in our c test case file as text file after saving these test case we have to load test case in source program but before these test case are going to load these test case can be generated using random search algorithm. Using concept of population we can generate no of test case from that test case we have to search best test case using search based algorithm. That can be used to cover maximum branch in source program for that purpose we use fitness function using fitness function block which produce value which is best value produce calculation is done to find best value also we can use mutation function to mutate the value so the best valued result is going to be produce by purposed system so the maximum branch coverage is possible with our system.

IV IMPLEMENTATION

In implementation section we are going to discuss how the actual purposed system work is Implemented. also how the algorithm is work and result are generated . First of all we can open an c/cpp file known as source file. after giving that file to our purposed system it will generate test case for correct source file. Now first save that test case and again load that test case in source file now doing same procedure using random search algorithm to generate no of test case which carried out correct test case and these test case can be save in c test case file.

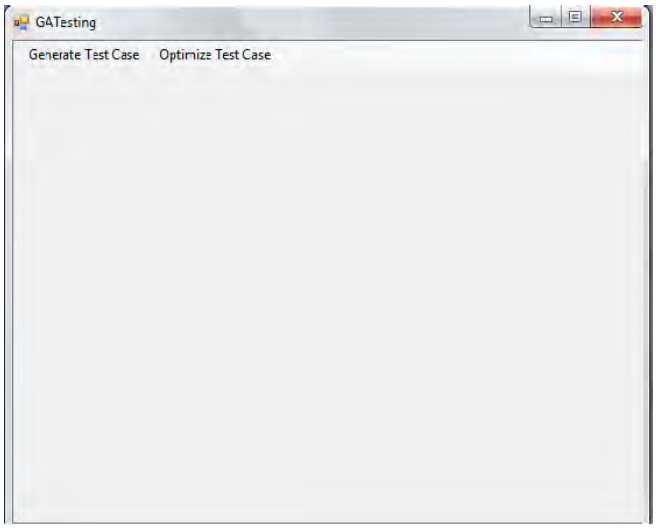

Fig no 3 Working with First Window

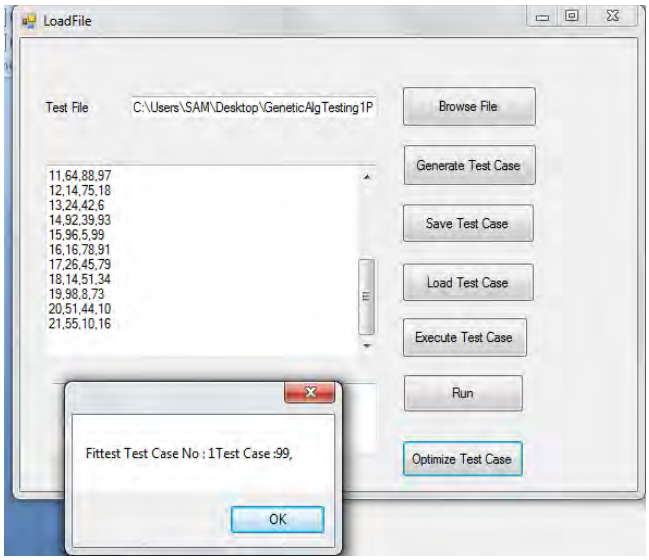

Fig No 4 fitness function for test case

Now from these test case I am going to search correct test case with help of genetic algorithm and fitness function these fitness function or operator is performing very important role in case of generation of test case because these fitness function can carried out an fittest integer value or fittest test case that can give us an optimal solution or optimal test case which cover maximum branch in source c program.

a) System Function:

The system function is how to generate test case of source program for the code is going to be cover. each condition in the code is required to be true at least one time I do framework testing for program unit .

The flow of the system is explain as follows

1)Analyze the tested source program with white testing initialize a population fitness function value and proper coding mode.

2)insert the point to tested that means test target branch which is involve in source program

3)testing data driven system running notes are write fitness function by point inserted evaluate the fitness function of each chromosome in the population. write result of every population to test case file.

4)produce the population from current population by applying mutation operation.

b) Discussion On Mutation Operator And Fitness Function:

After a new generator has been created a certain number of new solution are randomly chosen to experience mutation as the point insert method is adopted in testing program the insert parameter is changed when the new unit passes through the given route therefore the inserted parameter could be transferred to GA box as fitness function which is referenced of new unit.

c) Preparation of Source input file :

In this scenario the source input file can be give as input to the system my purposed system can split the source input file in to no of token and these no of token can be identified and using RS it is possible to generate test case now using my concept these token can be identified by variable.

My purposed system can be split these whole source file in to set of token now first system can create token as preprocessor directives #include refer to project specific source file. now standard c header file stdio.h can be check by system whether it is an part of source program or not after it will proceed to main prototype function where the variable are declared. Now these variable must be initialize with zero so that my system can use random function to generate an test case value for the variable declare inside main prototype. now by assigning a value to variable program proceed to check inside main function that where actual value is scan by scanf function these value can be identified by checking syntax of scanf function. After separating the variable check whether these function are declare integer or any other data type so according to these data type value can be assigned to variable randomly and population is created and these population can be test case which satisfies the condition for maximum branch coverage.

Since implementing software module as separate source and header file and control on source input file to generate an test case. also to identify an illegal variable is directly or indirectly it is necessary that variable is which type of variable after that I can create population for purposed system.

The purposed of this system is not only to generate an test case but also create an file that will display an maximum branch coverage. by performing data flow analysis of the parameter and variable by assigning individual value to variable declare inside main function .

A possible test strategy for determine the optimal solution in which to process my system module could be work as follow
1)Generate test case to cover the all statement from source input file.
2)continue to generate test case according to above criteria for which no test data search has been performed. this module of software would require to find out branch coverage data across all test data generation runs in order to achieve maximum branch coverage

V CONCLUSION

Finally in conclusion section there are various point that can be discus over here which can help me to provide an system with much more advantageous the conclusion can cover following point
1)Using random function it is easy to generate single test case but to generate no of test case that is population when I am going to implement an system with help GA it is necessary to create no of test case with some distance value between them so that it come to closer to result but using GA result is not so perfect.
2)To give an perfect result I am use GA with Mutation concept that can possible to give result but again it can not give an deterministic result because of population can't produce perfect result so to again giving perfect result it is necessary to use evolutionary approach that will provide perfect result
3)After using evolutionary algorithm and to generation maximum branch coverage and produce optimal solution it can be possible with the help of mutation and evolutionary approach for SBST with fitness function.
4)Fitness Function has been incorporated various condition required for test case. The use of GA to search for test case that satisfy the condition
5)Also FF and Mutation produce deterministic result.
6)Using evolutionary approach provide better experimental result.

ACKNOWLEDGMENT

I take this opportunity to express my profound gratitude and deep regards to my guide and Prof. Rahul S Gaikwad who has been very concerned and have aided for all the material essential for the preparation of this dissertation. He has helped me to explore this vast topic in an organized manner and provided me with all the ideas on how to work towards a research oriented venture. I am also thankful to Prof. Gosavi P. B., Prof. Dipak Pardhi sir. and Prof. Nilesh Wani for the motivation.

REFERENCES

[1]Andrea Arcuri "A Theoretical and Emprical analysis of Role of Test Sequence Length In software testing for Structural coverage"IEEE Trans.Vol 38 no 3 May/June 2012

[2]Yury Pavlov & G.Farser "Semi Automatic search based test generation"2012 IEEE inter.conf.

[3]Nikolai Kosmatov & n Williams "Structrual unit testing as service."2013 IEEE seventh international sympousim.

[3] J.H. Andrews, A. Groce, M. Weston, and R.G. Xu, "Random Test Run Length and Effectiveness," Proc. IEEE/ACM Int'l Conf.Automated Software Eng., pp. 19-28, 2008.(references)

[4] A. Arcuri, "Full Theoretical Runtime Analysis of Alternating Variable Method on the Triangle Classification Problem," Search Based Software Eng., pp. 113-121, 2009. (references)

[5] A. Richard torkar, "Insight Knowledge in Search Based Software Testing," Proc. Genetic and Evolutionary Computation Conf., pp. 1649-1656,2009. (references)

[6] A. Wasif Afzal "Theoretical Analysis of Local Search in Software Testing," Proc. Symp. Stochastic Algorithms, Foundations andApplications, pp. 156-168, 2009. (references)

[7] A. Robert fedult , "genetic algorithm and fitness function," Proc. IEEE Int'l Conf. Software Testing, Verification and Validation, pp. 205-214, 2010.

[8] A. Baresi and M Miraz "Evolutionary tes," Proc. IEEE Int'l Conf. Software Testing, Verification and Validation, pp. 469-478, 2009.

[9]Y.Dong J.Peng "Automatic generation of software test cases basedon improvedGeneticAlgorithm.