*International Journal of Engineering Science Invention (IJESI)*
*ISSN (Online): 2319 – 6734, ISSN (Print): 2319 – 6726*
*www.ijesi.org ||Volume 7 Issue 5 Ver VII || May 2018 || PP 110-116*

# Software Testing using Genetic Algorithm

Rajesh Kumar Pat[1]i, Nedunchezhian.T[2], Manmath Namth Dash[3]
*[1,3]Associate Professor, Department of Computer Science Engineering, Gandhi Institute For Technology (GIFT), Bhubaneswar*
*[2]Assistant Professor, Department of Computer Science Engineering, Gandhi Engineering College, Bhubaneswar*

***Abstract:****Testingisaprocessusedtoidentifythecorrectness,completenessandqualityofdevelopedcomputersoftware.Testing,apartfrom finding errors, is also used to test performance, safety, fault- toleranceorsecurity. Testingisthemostimportantqualityassurance measure for software. Testing is time consuming and laborious process. Therefore, techniques to automatic test data generation would be useful to reduce the cost and time. Software testing is an important and valuable part of the software development lifecycle.Duetotime,costandothercircumstances,exhaustive testing is not feasible that's why there is a need to automate the software testing process. Testing effectiveness can be achieved by the State Transition Testing (STT) which is commonly used inrealtime,embeddedandwebbasedtypeofsoftwaresystems. Theobjective ofthispaperistopresentanalgorithmbyapplying a Genetic Algorithm Technique, for generation of optimal and minimal test sequences for behaviour specification of software. Present paperapproach generatestestsequenceinorderto obtain the complete softwarecoverage.*

***Keywords:*** *Genetic Algorithms, Software Testing, Test Case Generation*

## I. Introduction

Testing is one of the most used software quality assessment methods.Therearetwoimportantprocesseswhentestingobjectorientedsoftwareareused.First,thesoftwarehastobeinitialized with a set of values. These values are used to set a number of variables that are relevant for the test case [3, 10]. The values of these variables define a single state from the possible set of states, the software can be determined. These values can either be a primitive value such as an integer or complex values such as an object. With the software testing initialized [9], itsmethod takes one or more software specification, defines the output ofthesoftwareandwhatisavalidinput.Sincethenumberofmoreobjectsasparameters,theseobjectsalsohavetobeinitialized.Todetermineifthetestcasepassedorfail,asoftwarespecification[8]hastobeused.Thenumberofpossiblestatessoftwaremayhave is exponential [12], it is impossible to test all of them. Software testing[9,12,19],isoneofthemajorandprimarytechniquesforachievinghighqualitysoftware.Softwaretestingisdoneto detect presenceoffaults[6,8],whichcausesoftwarefailure.However, software testing is a time consuming and expensivetask.

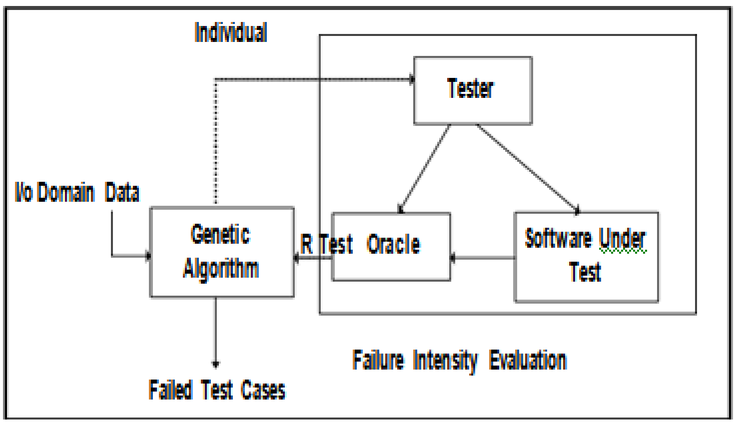


**Fig.1:**GAApproachtoFocusedSoftwareUsageTesting

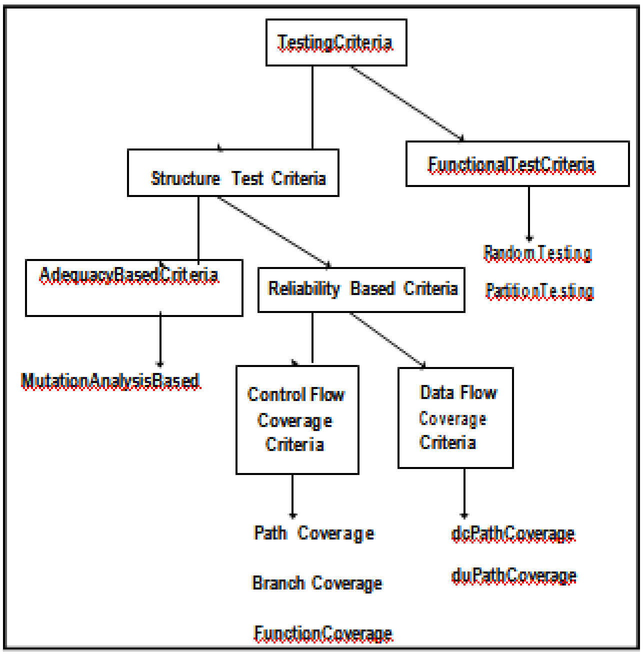


**Fig. 2:** General Classification Framework for Testing Criteria

This paper is design as follows Section I, Introduction, Section II,Key Research Concepts, Section III,Discussthe Conclusions. Finally, Section IV, Future Work of thepaper.

## II. Key Research Concepts

Our technique is based on the use of 'Genetic Algorithm' and 'Mutation Analysis'. These two concepts are the main focus of this paper.

### A. GeneticAlgorithms

Genetic Algorithms (GA) [4-5, 14], are search algorithmsbased onthenaturalselectionandgeneticreproductionasdescribedby CharlesDarwin.TheyareusedtofindsolutionstoOptimisation and search problems [12]. Genetic algorithms became popular when John Holland published the "Adaptation in Natural and ArtificialSystems",in1975andDeJongfinishedananalysisofthebehaviourofaclassofgeneticadaptivesystemsinthesa meyear. ThebasicideaofaGAistoencodethevaluesoftheparameters of an Optimisation problem in a chromosome [13, 22] which is evaluated by an objective function. GAs have been successfully applied to a wide range of applications, including Optimisation, scheduling, and design problems. GA has been applied in many Optimisation problems for generation test plants [10-11] for functionality testing, feasible test cases and in many otherareas. GA has also been used in model based test casegeneration.

Asshowninfig.3,thealgorithmstartsbyinitializingorrandomlygeneratingasetofchromosomes(population). Attheendofeach generation,eachchromosomeisevaluatedandmodifiedaccording to a number of generation operations in order to produce a new population[1,4].Thisprocessrepeatsuntilapredefinednumber of generations are computed.

Using genetic algorithms in test data generation for software testing is the process of identifying [10], a set ofprograminputdata,whichsatisfiesagiventestingcriterion[10-11].Intranslatingtheconceptsofgeneticalgorithmstotheproblemoftest-data[12] generation we perform the followingtasks:

- First of all we consider the population to be a set of test data.
- Findthesetoftestdatathatrepresentstheinitialpopulation. This set is randomly generated according to the format and type of data used by the program under test.
- Determiningthefitnessofeachindividualwhichisbasedon a fitness function that isproblem-dependent.
- Selecttwoindividualsthatwillbematedtocontributetothe next generation.
- Apply the crossover and mutationprocesses.

Theabovealgorithmwilliterateuntilthepopulationhasevolvedto formasolutiontotheproblem(satisfiesagiventestingcriterion), or until a termination condition issatisfied.

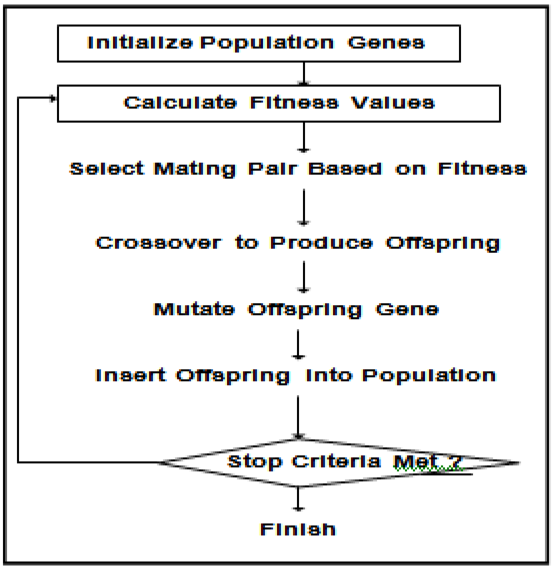


**Fig. 3:** Genetic Algorithm Flow Diagram

Basic Steps of a Typical Genetic Algorithm [5, 7]
Procedure GA Initialize population;
While termination condition not satisfied do
{
Evaluate current population;

$$p1 = \frac{\text{fitness}}{\sum_{i=0}^{\text{len(population)}} \text{fitness}_i}$$

**(c). TournamentSampling**
Uses the roulette wheel method to select two individuals. Then it
picks the one with the higher fitness value.

**(d). UniformSampling**
Selects and individual randomly from the population.

**(i). Stochastic RemainderSampling**
Firstcomputestheprobabilityofeachindividualbeingselected, p1 and its expected representation, Ɛ = p1*len(population). The expectedrepresentationisusedtocreateanewpopulationofthe samesize.Forexample,ifanindividualhasƐequalto1.7,itwill filloneposition inthenewpopulationandithasaprobabilityof 0.7tofillanotherposition.Afterthenewpopulationiscreated,the uniform method is used to select the individuals for mating.

**(ii). DeterministicSampling**
Computes Ɛ of each individual as in the stochastic remainder sampling [11]. A new population is created and filled with all individuals with Ɛ ≥ 1 and the remainder positions are filled by sortingtheoriginalpopulation'sfractionalpartsofƐandselecting the highest individuals on the list.

**(2). Crossover or Recombination**

Crossover is process where two or more chromosomes are combined to form one or more chromosomes. The idea behind crossover is that the offspring may be better than both parents. Crossover is normally done between two individuals, but more can be used.
There are many crossover algorithms [6]; some important algorithms are as described below:

Crossover; •
Mutation;
Set current population equal to be the newchild population; •
}
•
These primary operations include:

**(1). SelectionorReproduction** • A selection scheme is applied to determine how individuals are chosenformatingmethodbasedontheirfitness[13].Fitnesscanbedefinedasacapabilityofanindividualtosurviveandrep roducein an environment. Selection generates the newpopulationfromthe old one, thus starting a new generation. This operation assigns the reproduction probability to each individual based on the output of the fitness function. The individual based on the output of the fitness function. The individual with a higherrankingisgivenagreaterprobabilityforreproduction.As • a result, the filter individuals are allowed a better survival chance from one generation to the next.
Some of the selection schemes include:

**(a). Rank Schema**
Selects the best individuals of the populationevery time.

**(b). RouletteWheel**
Selectsindividualsaccordingtotheirfitnessvaluesascompared to the population. The probability [20] of an individual being picked is:
Uniform crossover will randomly select the parent where each gene should come from.
Evenoddcrossoverwillselectthegeneswithevenindexfrom parents A and the genes with odd index from parentB.
One point crossover will randomly select a position on the chromosome and all the genes to the left come from parent A and the genes to the right come from parent B.
Two points crossover ill randomly select two positions and pick the genes from parent A which have a greater index thanthesmallerpositionandasmallerindexthanthebiggest position. The remaining genes come from parentB.
PartialmatchcrossoverwillproducetwochildrenC1andC2. ItinitializesC1bycopyingthechromosomeoftheparentsA andC2bycopyingthechromosomeofparentB.Itwillthen randomly select a number of positions and swap the genes between C1 and C2 at those positions.
OrdercrossoverproducestwochildrenC1andC2.Itinitializes bycopyingthegenesoftheparentstochildrenanddeletingn genesrandomlyselectedfromeachoffspring.Itthenselects andintervalwithsizenslidesthegenessuchthattheinterval isempty.Itthenselectstheoriginalgenesinthatintervalfrom the opposite offspring[4].
CyclecrossoverproducestwochildrenC1andC2.Itinitializes C1 and C2 by copying the chromosomes of the parents A and B respectively. Then it selects n random positions and replaces the genes from C1withgenesfromparentsBinthose positions. The process is repeated for C2 with parentA.

The mutation operation is defined according to the structure of the chromosome. When the chromosome is stored in a tree, one possible mutation is to swap subtrees a shown in fig. 6.

**1. Purposes**

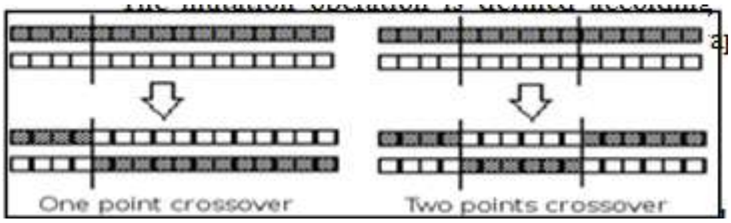

Fig. 4: One and Two Point's Crossover [6]

This operation is used to produce the descendants thatmake • up the next generation. This operation involves the following crossbreeding procedures:
They are provided a basis on which to decide whether something should be mutated.
They help in understanding and critiquing an existing set of mutation operators.

- Randomlyselecttwoindividualsasacouplefromtheparent generation.
- Randomly select a position of the genes, corresponding to thiscouple,asthecrossoverpoint.Thus,eachgeneisdivided into two parts.
- Exchange the first parts of both gene corresponding to the couple. Add the two resulted individuals to the next generation.

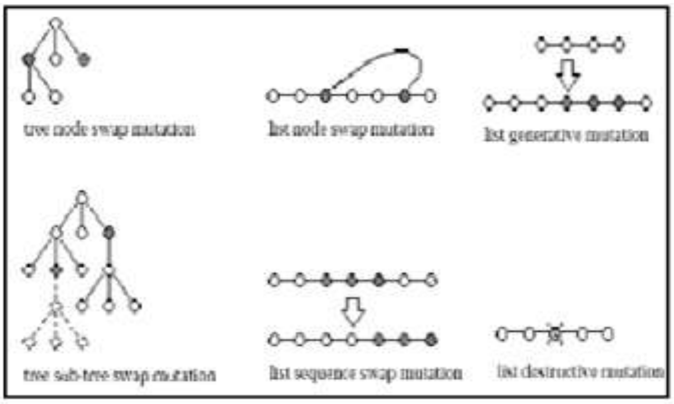


Fig. 6: Examples of mutation algorithms [3]

The main characteristics of GAs [2], are listed below:
- Concentrates on chromosomes with above averagefitness.
- Exploits information about a large number of values while processing a smallpopulation.
- Prevents search from stagnating at a localoptimum.
- They take advantage of old knowledge held in a population of solutions to generate new solutions with improved performance.

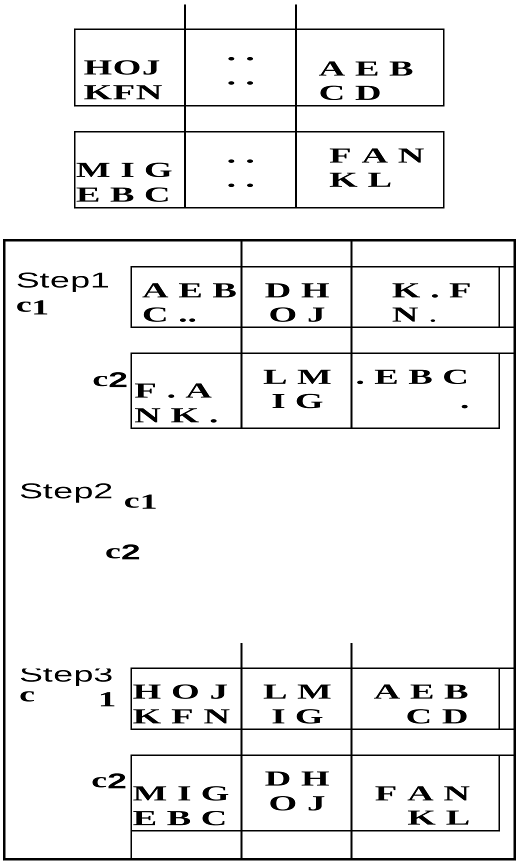


**Fig. 5:** Order Crossover Examples [4]

## B. Mutation

Mutation testing [24-25, 28] is a fault based testing technique used to find the effectiveness of test cases. It is a powerfulandcomputationallyexpensivetechniquetofindtheadequacyoftestcases[22,26].Highqualitysoftwarecanno tbedonewithouthigh quality testing.

Mutationtestingmeasureshow“good”ourtestsarebyinserting faults into the program under test. Each fault generates a new program, a mutant, which is slightly different from the original. The idea is that the tests are adequate if they detect all mutants [16-17].

This operation picks a gene at random and changing its state according to the mutation probability [12]. The purpose of the mutation operation is to maintain the diversity in generation to prevent premature convergence to a local optimal solution. The mutation probability is given intuitively since there is no define way to determine the mutation probability.

Three basic operations are described below:

- Flipmutator[18],willchangeasinglegeneofthechromosome to a random value according to the range specified by the alleles(allelesisanalternativeformofagenethatislocated at a specific position on a specificchromosome).
- Swapmutatorwillrandomlyswapanumberofgenesofthe chromosome.
- Gaussian mutator will pick a new value around the current value using a Guassian distribution[3].

## III. Conclusion

This paper introduces a genetic algorithm approach to software usage testing that is used to explore the space of input data and identify and focus on regions that cause failure. Analysis of the examplesinthispaperdemonstratesthatgeneticalgorithmscanbeusedasatooltohelpasoftwaretestersearch,locate,andi solatefailuresinasoftwaresystemtesting.Theuseofgeneticalgorithmssupportsautomatedtestingandhelpstoidentifyth osefailuresthat are most severe and likely to occur for theuser.

## IV. FutureWork

Despiteofthebigimprovementachievedbyevolutionarytesting [2], there are many improvements that should increase this improvementevenfurther.Somepossibleimprovements[6]and research directions are presented below:

### A. Efficiency

The efficiency of the algorithm can be improved by combining the genetic algorithm and Auto test into a single system. At the moment, every time Auto test is invoked from the genetic algorithm, it has to load and parse the class under test.

### B. ReusingStrategies

The strategies evolved by the genetic algorithm may be reusedwhen evolving a new strategy [7] for a new class.

### C. StaticAnalysis

This system uses a vary naive static analysis technique; a more advancedtechniquemaybeabletoimprovetheadaptationofthe evolutionary strategy.

### D. CodeMetrics

At the moment the information about the complexity of the methodsbeing testedisnottakeninto account.Wheninitializing the testing strategies, code metrics [22] could be used to have a smarterinitialization.

### E. Evolution ofStrategies

Byevolvingatestingstrategyforalongertimemayincreasethe number of faults found, especially for classes [5, 9] with a low number of faults.

There are other problems as flag and enumeration variables and unstructured control flow. Additional researches are required to overcome these problems.

One of the major difficulties [10, 13] in software testing is the automatic generation of test data that satisfy a given adequacy criterion. To solve this difficult problem there were a lot of research works, which have been done in the past. Perhaps the most commonly encountered are random test-data generation, symbolic(orpath-oriented)test-datagenerationbasedongenetic algorithm [12, 14].

## References

[1]. Goldberg,D.E.,“Genetic Algorithms in Searchof Optimisation”,1989.

[2]. B.Korel,“Dynamicmethodforsoftwaretestdatageneration,” Software Testing, Verification & Reliability, vol. 2, 1992, pp. 203-213.

[3]. Alan C. Schultz, John J. Grefenstette, aid Kenneth A. De Jong,“Test And Evaluation by Genetic Algorithms”, IEEE, 1993.

[4]. Jefferson Offutt, W. Michael Craft,“Using Compiler Optimisation Techniques to Detect Equivalent Mutants”, TheJournalofSoftwareTesting,VerificationandReliability, 4(3), pp. 131–154, 1994.

[5]. Painton,L.,Campbell,J.,"GeneticalgorithmsinOptimisation of system reliability", IEEE Trans. Reliab. 44 (2), pp.172– 178, 1995.
[6]. Michalewicz, Z.,"Genetic Algorithms, Data Structures Evolution Programs", Verlag, Heidelberg, Berlin, Third Revised and Extended Edition, 1999.
[7]. Tian, J.,"Measurement and continuous improvement of softwarereliabilitythroughoutsoftwarelife-cycle",J.Syst. Software 47,1999.
[8]. RoyPPargas,MaryJeanHarrold,RobertRPeck,"TestData GenerationUsingGeneticAlgorithms",JournalofSoftware Testing, Verification andReliability,1999.
[9]. W. Caarls.,"Genetic algorithm visualisation. Master's thesis, Faculty of Science", University of Amsterdam, The Netherlands, September2002.
[10]. S. Yang.,"Statistics-based adaptive non-uniform crossover forcrossoverforgeneticalgorithms",InJ.Bullinaria,editor, Proceedings of the 2002 UK Workshop on Computatiorial lntelligence (UKCI-02), pp. 201-208, 2002.
[11]. P. McMinn,"Search-Based Software Test Data Generation: A Survey", Software Testing, Verification and Reliability, vol. 14, No. 2, pp. 105-156,2004.
[12]. Phil.McMinn,"Search-BasedSoftwareTestDataGeneration: A Survey", Software Testing, Verification and Reliability, Vol. 14, No. 3, pp. 212-223,2004.
[13]. M.R.Girgis,"AutomaticTestDataGenerationforDataFlow Testing Using a Genetic Algorithm", Journal of Universal Computer Science, vol. 11, No. 6, pp. 898-915,2005.
[14]. B. Antonia,"Software Testing Research: Achievements, Challenges,Dreams",in2007FutureofSoftwareEngineering: IEEE Computer Society, 2007.
[15]. S. Bandyopadhyay, S. K. Pal,"Classification and Learning Using Genetic Algorithms", Application in Bioinformatics and Web Intelligence. Berlin, Germany: Springer,2007.
[16]. M.Kay,"XSLTransformations(XSLT)version2.0",W3C, W3C Recommendation, (2007), [Online] Available: http:// www.w3.org/TR/2007/REC-xslt20-20070123/.
[17]. KimS.,ClarkJ.A.,McdermidJ.A.,"Classmutation:mutation testing for object-oriented programs", Proc. Net. Object Days, October 2000, [Online] Available: http://wwwusers.cs.york.ac.uk/~jac/papers/ClassMutation.pdf, accessed, March2008.
[18]. H. Shahriar,"Mutation-based testing of buffer overflows, SQLinjections,andformatstringbugs",Ph.D.dissertation, University of Queen, Aug.2008.
[19]. H. Shahriar, M. Zulkernine,"MUSIC: mutation-based SQL injection vulnerability checking", in Proceedings of the8th International Conference on Quality Software (QSIC '08), Aug. 2008, pp.77–86.
[20]. Yong Chen, Yong Zhong, Tingting Shi, Jingyong Liu, "ComparisonofTwoFitnessFunctionsforGA-basedPath- Oriented Test Data Generation", 2009 Fifth International Conference on Natural Computation, IEEE,2009.
[21]. Y. S. Ma, Y. R. Kwon, S. W. Kim,"Statistical investigation on class mutation operators", ETRI Journal, Vol. 31, No.2, pp. 140–150, 2009.
[22]. M.Singh,S.Mishra,"Mutantgenerationforaspectoriented programs," Indian Journal of Computer Science and Engineering, Vol. 1, No. 4, pp. 409–415,2010.
[23]. U.Praphamontripong,J.Offutt,"ApplyingmutationtestingtoWebapplications",inProceedingsofthe3rdInternationalConferenceonSoftwareTesting,Verification,andValidation Workshops (Mutation 2010). IEEE ComputerSociety, Apr. 2010, pp. 132–141.
[24]. "QuantitativeevaluationofmutationoperatorsforWSBPEL compositions", in Proceedings of the 3rd International ConferenceonSoftwareTesting,Verification,andValidation Workshops (Mutation 2010). IEEE Computer Society,Apr. 2010, pp. 142–150.
[25]. Y. Jia, M. Harman,"An Analysis and Survey of the Development of Mutation Testing", IEEE Transactions of Software Engineering, Vol. To appear,2010.
[26]. H.Zhang,"MutationoperatorsforC++",Feb.2010.[Online] Available: http://www.people.cis.ksu.edu/~hzh8888/mseproject.htm.
[27]. L. Madeyski, N. Radyk,"Judy – A mutation testing toolfor Java," IET Software, Vol. 4, No. 1, pp. 32–42,2010.
[28]. F. Lonetti, E. Marchetti,"X-MuT: a tool for the generation of XSLT mutants", in Proceedings of the Seventh Interna- tional Conference on the Quality of Information and CommunicationsTechnology.IEEEComputerSociety,2010, pp. 280–285.