Prioritizing Test Cases in Regression testing using Fault Based Analysis

Ashima Singh

**IT Department, MM University, Mullana
133002,India**

Abstract

Testing is an integral part of any software development lifecycle. It takes considerable amount of time and capital to generate test cases and apply testing. Genetic Algorithms are proving to be great tool in optimizing software testing. This paper uses the concept of Genetic algorithms in optimizing software testing. In this paper, we have analysed genetic algorithms and studied their effectiveness to find the faults and time overhead-based criteria to -prioritize test cases. The proposed approach is providing the solution of test cases sequencing as well as reduction by using an intelligent dynamic approach. The proposed system will generate the test cases based on the priorities, which are assigned by the algorithm to test cases on the basis of some intelligent operations. A cumulative mutation probability (CMP) metric is used to determine the effectiveness of the new test case orderings

Keywords: **Software Testing, Regression Testing, Genetic, offspring, Prioritization.**

1. Introduction

Software Testing means computing the system with purpose of finding errors. It is an application for a concerted action of a system under controlled conditions and evaluating the results. Once system has been developed, it must be tested before it implementation. It is oriented towards Error-detection.

Software testing is one element of a broader topic that that is often referred to as verifying and validating that a software application or program. Software testing is useful for finding the defects, fundamental weakness in the application code that must be improved or checked**.**

Software testing has three main purposes: verification, validation, and defect**:**

- The process of verification confirms that software meets its specifications. It ensures that software correctly implemented for specific function.
- Whereas the process of validation ensures that the software meets the business requirements. It provides the traceable activities to customers.
- A defect is inconsistency among the expected and actual result. The defect's ultimate source may be traced to a fault introduced in the specification, design, or development phases.

In development of software system, cost of testing a program is associated **[1]**. Tester has to write test plan and test cases, to set up the proper equipment, systematically execute the test cases, and follow up on problems that are identified also try to remove most of the faults. For faults that are not discovered and removed before the software has been shipped, there are costs. Some of these costs are monetary, and some could be significant in less tangible ways. Customers can lose faith in our business and can get very angry. They can also lose a great deal of money if their system goes down because of our defects. And, software development organizations have to spend a great deal of money to obtain specific information about customer problems and to find and fix the cause of their failures.

To minimize the costs associated with testing and with software failures, a goal of testing must be to uncover as many defects as possible with as little testing as possible. In other words, we want to write test cases that have a high likelihood of uncovering the faults that are the most likely to be observed as a failure in normal use. It is simply impossible to test every possible input-output combination of the system; there are simply too many permutations and combinations. As testers, we need to consider the economics of testing and strive to write test cases that will uncover as many faults in as few test cases as possible.

2. Related Work

The process of software maintenance is an activity which includes enhancements, error corrections, optimization and deletion of obsolete capabilities. These modifications in the software may cause the software to work incorrectly and may also affect the other parts of the

Copyright (c) 2012 International Journal of Computer Science Issues. All Rights Reserved.

software. In order to prevent this Regression testing is performed.

Regression testing is used to revalidate the modifications of the software. Regression testing is an expensive process in which the test suites are executed ensuring that no new errors have been introduced into previously tested code and after the modification the software is working properly.

Regression testing is done in the maintenance phase of the software development life cycle to retest the software for the modifications it has undergone. Approximately 50% of the software cost is involved in the maintenance phase so researchers are working hard to come up with best results by developing new Regression Testing techniques.

The most important concerns in the testing phase are time and cost. Test Case Prioritization is a classification type of Regression Testing that helps us to make the process of testing a cost and time effective task. It also increases the efficiency of the testing procedure by prioritizing the test cases according to desired criteria.

Akira K. Onoma, Wei-Tek Tsai, Mustafa H. Poonawala, and Hiroshi Suganuma [1998],"Regression Testing in an Industrial Environment"- In this paper author's discussed the issues such as test case revalidation, test execution, failure identification, fault identification, modification dependency, fault mitigation and test case dependency are essential for an industrial environment in applying regression testing. Most of these issues are easy to address if one is dealing with small programs, but in a large software house where multiple large programs are being developed and maintained, they suddenly become complicated and costly. Some of these issues are also related to general software testing, however, these problems become acute in regression testing because test cases are repeatedly exercised in case of regression testing.

Mark Last, Shay Eyal, and Abraham Kandel proposed a new [2005],"Effective Black Box Testing with Genetic Algorithms"- proposed a GA-based approach to generate effective black-box test cases. From the case study, they conclude that the Fuzzy- Based Age Extension of Genetic Algorithm (FAexGA) is much more efficient for this problem than the two other evaluated algorithms (SimpleGA and GAVaPS). In this paper, they introduced a new, computationally intelligent approach to generation of effective test cases based on a novel, Fuzzy-Based Age Extension of Genetic Algorithms (FAexGA). The basic idea was to eliminate bad test cases that are unlikely to expose any error. The promising performance of the FAexGA based approach was demonstrated on testing a complex Boolean expression.

Xiaofang Zhang, Changhai Nie, Baowen Xu, Bo Qu [2007], "Test Case Prioritization based on Varying Testing Requirement Priorities and Test Case Costs"- This paper discusses prioritization of test cases in regression testing. It schedules test cases in order of precedence that increases their ability to meet some performance goals, such as code coverage, rate of fault detection. In this paper, basing on varying testing requirement priorities and test case costs, we present a new, general test case prioritization technique and an associated metric. The case study illustrates that the rate of "units-of-testing-requirement-priority-satisfiedper-unit-test-case-cost" can be increased, and then the testing quality and customer satisfaction can be improved.

Gaurav Duggal, Bharti Suri [2008],"UNDERSTANDING REGRESSION TESTING TECHNIQUES"- described Regression testing is done in the maintenance phase of the software development life cycle to retest the software for the modifications it has undergone. Approximately 50% of the software cost is involved in the maintenance phase so researchers are working hard to come up with best results by developing new Regression Testing techniques so that the expenditure made in this phase can be reduced to some extent. This paper discussed Regression Testing techniques and further classified each one of them respectively as explained by various authors, explaining Regression Test Selection and Test Case Prioritization in detail with Search Algorithms for Test Case Prioritization. Through this paper author tried to, explain the complete structure of Regression Testing, areas of Regression Testing to make researchers understand its importance and scope and motivate new researchers who are planning to start their research "to work on it".

Chen L., Wang Z., Xu L. [2010],"Test Case Prioritization for Web Service Regression Testing"- proposed a dependence analysis based test case prioritization technique for Web Service regression testing. First, they analyzed the dependence relationship using control and data flow information in an orchestration language: WS-BPEL. Then they construct a weighted graph. After that, they prioritize test cases according to covering more modification-affected elements with the highest weight. Finally authors conduct a case study to illustrate the applicability of method.

Hyunsook Do, Siavash Mirarab[2010], "The Effects of Time Constraints on Test Case Prioritization: A Series of Controlled Experiments"- Several series of experiments are conducted to assess the effects of time constraints on the costs and benefits of prioritization techniques. Results of different experiments are:

- Manipulates time constraint levels and shows

Copyright (c) 2012 International Journal of Computer Science Issues. All Rights Reserved.

that time constraints do play a significant role in determining both the cost-effectiveness of prioritization and the relative cost-benefit trade-offs among techniques.

- Replicates the first experiment, controlling for several threats to validity including numbers of faults present, and shows that the results generalize to this wider context.
- Manipulates the number of faults present in programs to examine the effects of faultiness levels on prioritization and shows that faultiness level affects the relative cost-effectiveness of prioritization techniques.
- If they are considered together the results have suggestions about when and when not to prioritize, techniques to be employed and how differences in testing processes may relate to prioritization cost-effectiveness.

Chen Zhang, Zhenyu Chen, Zhihong Zha[2010], "An Improved Regression Test Selection Technique by Clustering Execution Profiles"-This paper presents a new regression test selection technique by clustering the execution profiles of modification traversing test cases. Cluster analysis can group program executions that have similar features, so that program behaviours can be well understood and test cases can be selected in a proper way to reduce the test suite effectively. An experiment with some real programs is designed and implemented. The experiment results show that the approach can produce a smaller test suite with most fault-revealing test cases in comparison with existing selection techniques.

Anoj Kumar, Shailesh Tiwari, K. K. Mishra[2010], "Generation of Efficient Test Data using Path Selection Strategy with Elitist GA in Regression Testing"- It presents a combined approach by which the stated problems are resolved in effective manner. By this approach, tester can identify the appropriate paths for test case execution and also generate efficient test data using elitist version of GA. The proposed approach enables tester to execute the test cases in order to increase their effectiveness to find faults taking minimum efforts. It can be used in regression testing to choose an appropriate subset of test cases by using elitist GA, among a previously run test suite for a software system, based on the information about the modifications made to the system for enhancement.

3. Regression Testing

Regression Testing is an important strategy for reducing side effects. We run regression testing every time software experiences a change in form of bug fixes or some additional functionality. It is done to ensure that code had not an adverse effect to the other module or any existing functions and it may not have produced any defect.

The regression test suite contains three different classes of test cases:

- A representative sample of tests that will exercise all software functions.
- Additional test that focuses on software function that are likely to be affected by change.
- Test that focus on components that have been changed.

A subset of the regression test cases can be set aside as Smoke tests. A smoke test is a group of test cases that establish that the system is stable and all major functionality is present and works under "normal" conditions.

Smoke tests are often automated, and the selections of the test cases are broad in scope. The smoke tests might be run before deciding to proceed with further testing (why dedicate resources to testing if the system is very unstable). The purpose of smoke tests is to demonstrate stability, not to find bugs with the system.

The most crucial phase in the software development life cycle is maintenance phase, in which the development team is supposed to maintain the software which is delivered to the clients by them. Software maintenance results for the reasons like error corrections, enhancement of capabilities, deletion of capabilities and optimization.
Regression testing is defined as "The process of retesting the modified parts of the software and ensuring that no new errors have been introduced into previously tested code".

The various types of techniques for regression testing are:

- **Retest all:** method is one of the conventional methods for regression testing in which all the tests in the existing test suite are re-runned. So the retest all technique is very expensive as compared to techniques which will be discussed further as regression test suites are costly to execute in full as it require more time and budget.
- **Regression test selection:** approaches attempt to reduce the cost of regression testing by selecting some appropriate subset of the existing test suite .Test selection techniques normally use the source code of a program to determine which tests should be executed during the regression testing stage .
- **Regression test prioritization** techniques attempt to order a regression test suite so that those tests with the highest priority, according to some established criterion, are executed earlier in the regression testing process than those with

Copyright (c) 2012 International Journal of Computer Science Issues. All Rights Reserved.

lower priority. By prioritizing the execution of a regression test suite, these methods hope to reveal important defects in a software system earlier in the regression testing process.

- **Hybrid approach:** also known as regression test distribution is another alternative that can make regression testing more practical by more fully utilizing the computing resources that are normally available to a testing team.

4. Proposed Approach

4.1 A New Prioritization Technique

Earlier work may take long time (may be month or year) depending on the size of the test suite and how long each test case takes to run. However through the use of an effective prioritization technique, testers can re order the test cases to obtain an increased rate of fault detection.

The technique presented in this paper presented a new regression test suite prioritization algorithm that prioritizes the test cases with the goal of maximizing the number of faults that are likely to be found during the constrained execution.

4.2 The Algorithm

1. Accept the N Test Cases in form of TestCost Matrix
2. Define the Initial Population Size called PopSize
3. Generated the Random Population Set to represent the possible test sequences
4. Define Fitness Function
5. Select two Random Parents from Population Set
6. Perform two point crossover to generate new Child
7. Perform Mutation Operation
8. Add newly generated child to population
9. Return Optimized Test Sequence

The proposed approach is the try to reduce the test cases and assigning a new prioritization sequence. We need to define a database to maintain all the test cases respective to the project. The data will contain different kind of test respective to the criticality level. It will also define the position of the test cases in the data flow over the object. It also define either it is a function test or non function test.

Once all the test cases are defined the next work is assign the priorities to these test cases. The prioritization should be assigned according to the criticality of the test as well as the code on which the test is occurred. It also defines how frequent the test is. After considering an initial test cases sequence is generated.

We need to define the event that can affect the available code or the related test cases. With each event we define the affected test cases. The test cases affection is represented as use case. It also defines as the event the particular test cases will be required to perform or not. If it is required it will check weather it will be uses in same or some modification is required. After the use case is assigned to the available test cases the next work is to assign a new sequence of test case implementation. This work will be performed dynamically by keeping the existing test cases in mind as well as by observing the criticality level as well the use cases of the particular test case.

The fundamental mechanism consists of the following stages :

1. Generate randomly the initial population.
2. Select the chromosomes with the best fitness values.
3. Recombine selected chromosomes using crossover and mutation operators.
4. Insert offspring into the population.
5. If a stop criterion is satisfied, return the chromosome with the best fitness.
6. Otherwise, go to Step 2.

5. Computational Experimentation

We are defining a new approach to assign the priorities to the test cases dynamically while performing the regression testing using genetic approach. It will reduce the cost thus assigning a new prioritization sequence. For this purpose we have used MATLAB simulator because it provides high-level technical computing language and interactive environment for algorithm development, data visualization, data analysis, and numeric computation.

To show the presented work we have estimate the cost respective to number of test cases. The overall cost in case of different test cases is given as under.

5.1 Genetics based Test Cost Analysis (100 Generations)

We have analysed these test cases based on the cost of the test case. The cost depends on the occurrence and the detection of the software fault in a particular module. Here we have assigned the test cost in different ways to perform the analysis. The basic parameters defined while performing the Genetic Algorithm are:

Parameter	Value
Number of Test Case	10
Test Cost	0 to 1
Generations	100

Copyright (c) 2012 International Journal of Computer Science Issues. All Rights Reserved.

Crossover	PMX
Optimization Criteria	Minimization

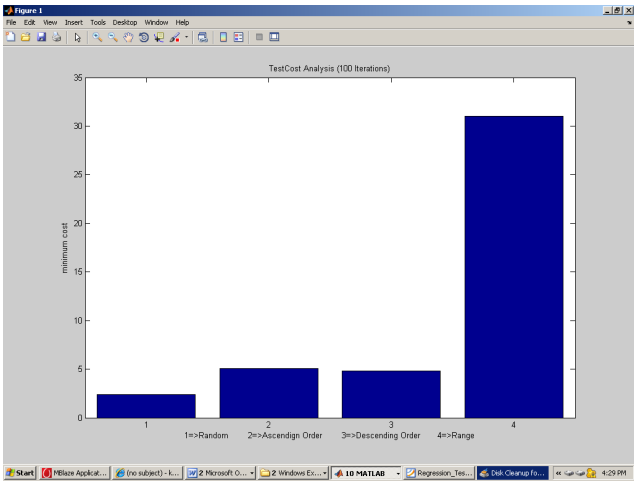

Figure 1. Test Cost Analysis (All Methods/100 Generations)

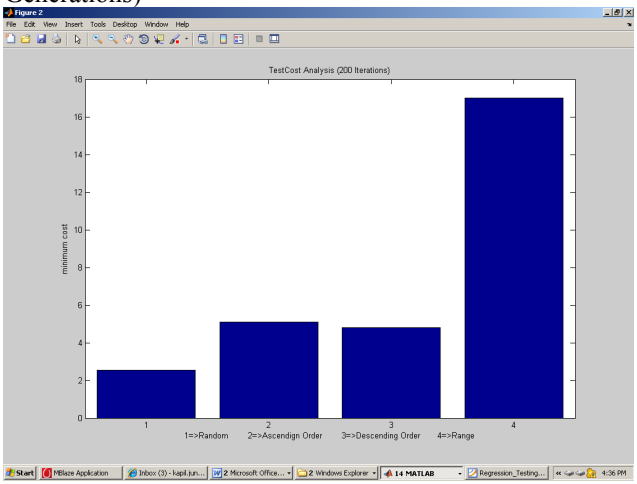

Figure 2. Test Cost Analysis (All Methods/200 Generations)

5.2 Cost Evaluation for Random Cost Assignment

As the general case we have assigned the random cost to each test case and perform the analysis based on this random cost assignment. The output driven based on this assignment is shown as under.

a) The obtained Test Sequence of this random cost assignment is given as
3 9 4 1 10 7 8 5 6 2
b) The Process cost driven from the genetic on initial cost assignment is given as
Process Cost = 11.0547
c) The cost driven after implementation of optimized test sequence is given as
Test Cost = 1.3207

5.3 Cost Evaluation in Ascending Order

As the general case we have assigned the cost in increasing order of test cost to each test case and perform the analysis based on this cost assignment. The output driven based on this assignment is shown as under.

a) The obtained Test Sequence of this random cost assignment is given as
7 2 1 5 10 6 9 3 4 8
b) The Process cost driven from the genetic on initial cost assignment is given as
Process Cost = 9.5599
c) The cost driven after implementation of optimized test sequence is given as
Test Cost = 5.3

5.4 Cost Evaluation in Descending Order

As the general case we have assigned the cost in decreasing order of test cost to each test case and perform the analysis based on this cost assignment. The output driven based on this assignment is shown as under.

a) The obtained Test Sequence of this random cost assignment is given as
9 3 7 1 10 2 8 5 6 4
b) The Process cost driven from the genetic on initial cost assignment is given as
Process Cost = 12.9098
c) The cost driven after implementation of optimized test sequence is given as
Test Cost = 4.9

1. Cost Evaluation on Range Based

As the general case we have assigned the cost in range between 1 and 10. Here the cost assignment is developer assisted and the analysis is performed on the basis on this cost assignment. The output driven based on this assignment is shown as under.

a) The obtained Test Sequence of this random cost assignment is given as
7 3 4 6 2 8 10 1 9 5
b) The Process cost driven from the genetic on initial cost assignment is given as
Process Cost = 10.4791
c) The cost driven after implementation of optimized test sequence is given as
Test Cost = 24

6. Determining Test Suite Effectiveness

6.1 APFD Average Percentage of Fault Detection (APFD) Metric

To quantify the goal of increasing a subset of the test suite's rate of fault detection, i use a metric called APFD developed by Elbaum et al. that measures the average rate of fault detection per percentage of test suite execution. The APFD is calculated by taking the weighted average of the number of faults detected during the run of the test suite. APFD can be calculated using a notation:

Copyright (c) 2012 International Journal of Computer Science Issues. All Rights Reserved.

Let T -> The test suite under evaluation program under test P
n -> The total number of test cases and
TFi -> The position of the first test in T that exposed fault i.

$$APFD = 1 - \frac{TF_1 + TF_2 + \ldots\ldots + TF_m}{Nm} + \frac{1}{2n}$$

So as the formula for APFD shows that calculating APFD is only possible when prior knowledge of faults is available. APFD calculations therefore are only used for evaluation.

6.2. Experimentation and analysis

Following table shows the number of faults detected by a test case in the test suite and total time taken by each test case.

	T1	T2	T3	T4	T5	T6	T7
F1	X	X				X	
F2				X			
F3							X
F4					X	X	
F5			X		X		
F6				X		X	
F7				X			X
Number of Faults	1	1	1	3	2	3	2
Time	5	7	11	4	10	12	6

6.3 APFD VALUE FOR NON PRIORITIZED TEST SUITE:

m= number of faults=7
n= number of test cases=7
Test sequence= T1,T2,T3,T4,T5,T6,T7

Putting values in formula:

APFD = 1 – (1+4+7+5+3+4+4)/7*7 + 1/(2*7)
= 0.50

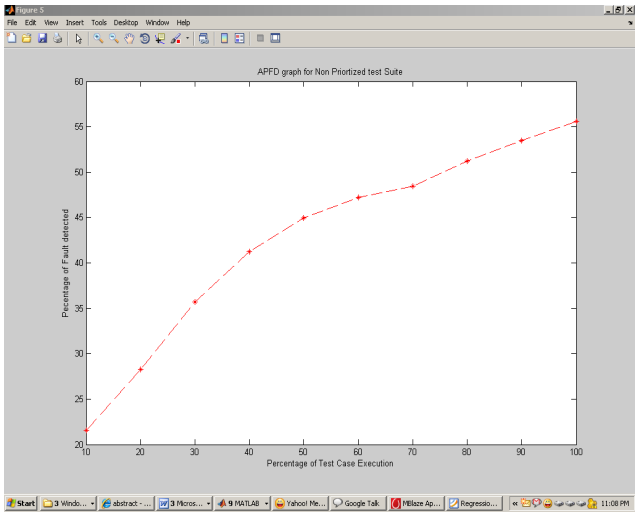

Figure 3 APFD Analysis (Basic Test Sequence)

6.4 *APFD VALUE FOR PRIORITIZED TEST SUITE:*

The prioritized test sequence obtained after applying Genetic Algorithm and its operators crossover and mutation T4 T7 T1 T5 T3 T2 T6
Calculation on putting values:-
APFD = 1 – (3+1+2+4+5+1+1)/7*7 + 1/(2*7)
=0.72

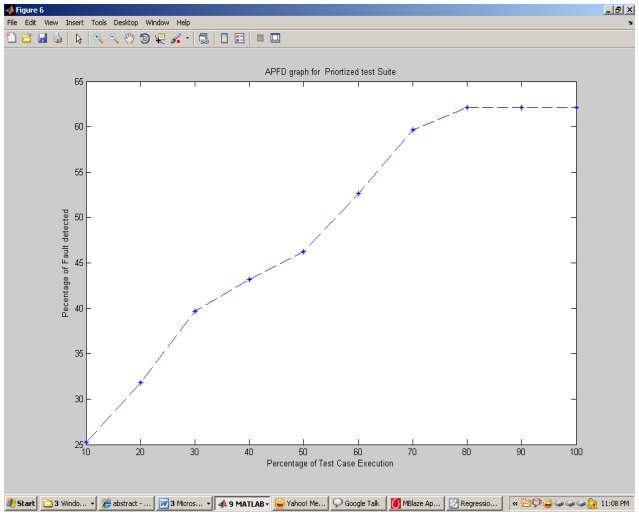

Figure 4. APFD Analysis (Prioritized Test Sequence)

6.5 Analysis of APFD

Results calculated by APFD proves that prioritized test sequence is more effective in finding out the faults in less time.

Copyright (c) 2012 International Journal of Computer Science Issues. All Rights Reserved.

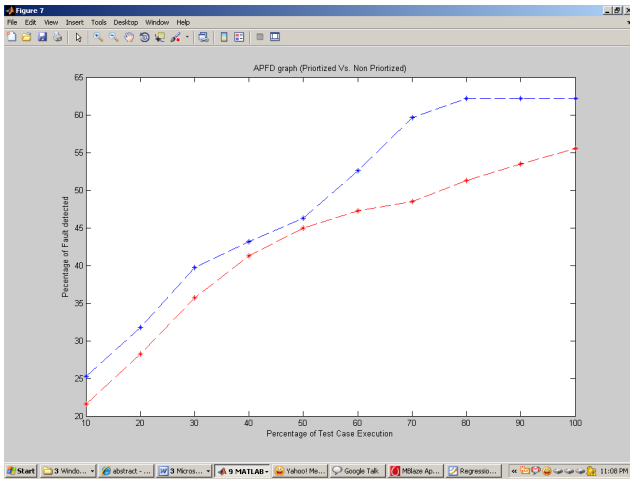

Figure 5. Comparison Graph(Prioritized and non Prioritized Sequence)

7. Discussion and Conclusion

In this present work we are defining a new approach to find the optimal test sequence dynamically while performing the regression testing. From this complete work we can conclude the genetic can be used effectively to perform the work on test case generation But it can be used effectively if there is large number of test cases and large number of possible test sequences. More the number of possible test sequence more effective the result will be. The conclusion driven here are

- In this work we find the effectiveness of genetic optimization for regression testing.
- The obtained test sequence is less costly and less critical then other.
- This paper proposed the Genetic Algorithm for prioritization of test cases to improve the regression Testing. Analysis is done for prioritized and non prioritized cases with the help of APFD metric .Graph proved that prioritized case is more effective.
- We can use some other optimization approaches such as ACO, Swarm based approach etc. to generate the prioritized test sequence.

If we have a large test suite then we can implement the clustering to categorize the faults and then perform the cluster based prioritization approach.

References

[1] Gregory M. Kapfhammer [1968]-“Software Testing”.
[2] D. Richardson, O. O’Malley and C. Title [1989], “Approaches to specification-based testing”.
[3] Wong W., Agrawal. H.[1997],” A Study of Effective Regression Testing in Practice”.
[4] Akira K. Onoma, Wei-Tek Tsai, Mustafa H. Poonawala, and Hiroshi Suganuma [1998],"Regression Testing in an Industrial Environment".
[5] G. Rothermel, M. J. Harrold, J. Ostrin, and C. Hong [1998],“An empirical study of the effects of minimization on the fault detection capabilities of test suites”.
[6] K. Onoma, W.-T. Tsai, M. Poonawala, and H. Suganuma [1998], “Regression testing in an industrial environment”.
[7] M. J. Harold, J. A. Jone, T. Li, and D. Liang [2001], “Regression test selection for java software”.
[8] Elbaum, Malishevsky, Rothermel [2002],"Test case prioritization: a family of empirical studies".
[9] Y. Chen, R. Probert, and D. P. Sims[2002], “Specification-based regression test selection with risk analysis”.
[10] David Leon, Andy Podgurski [2003],"A Comparison of Coverage-Based and Distribution-Based Techniques for Filtering and Prioritizing Test Cases".
[11] Mark Last, Shay Eyal, and Abraham Kandel proposed a new [2005],"Effective Black-Box Testing with Genetic Algorithms".
[12] H. Srikanth [2005], “Requirements-based test case prioritization”.
[13] Xiaofang Zhang, Changhai Nie, Baowen Xu, Bo Qu [2007], "Test Case Prioritization based on Varying Testing Requirement Priorities and Test Case Costs".
[14] Gaurav Duggal, Bharti Suri [2008],"UNDERSTANDING REGRESSION TESTING TECHNIQUES".
[15] C. Jones [2008], “Software quality in 2008: A survey of the state of the art”.
[16] Srivastava P., Kim P.[2009], ”Application of Genetic Algorithm in Software Testing”.

Copyright (c) 2012 International Journal of Computer Science Issues. All Rights Reserved.