Test Case Optimization and Prioritization of Web Service Using Bacteriologic Algorithm

Gaurav Raj, Dheerendra Singh and Ishita Tyagi

Abstract Regression testing, testing is done on the changes made in existing software to check whether the existing software is working properly or not after the changes has been done. Therefore, retesting is performed to detect the new faults found. This type of testing is performed again and again after the changes have been made in the pre-existing software. Various methods are used for test case reduction and optimization for a web service. Regression testing creates a large number of test suites which consumes a lot of time in testing and many other problems are faced. Therefore, some technique or method should be used so that number of test cases are reduced and also test cases can be prioritized keeping in mind the time and budget constraints. The test case reduction and prioritization need to be achieved depending on various parameters such as branch coverage and also on basis of fault coverage etc. Therefore, this paper discusses about the analysis of the code of a web service and the technique used to analyze a web service based on branch or code coverage and also the fault detection for test case reduction and prioritization is bacteriologic algorithm (BA). The test cases generated and also other requirements are mapped with the branch coverage and fault coverage of the code of the web service.

Keywords Test case · Test suite · Bacteriologic algorithm · Web service
Regression testing · Test case reduction

G. Raj (✉)
Punjab Technical University, Kapurthala, Punjab, India
e-mail: graj@amity.edu

D. Singh
Computer Science and Engineering Department, Chandigarh College of Engineering and Technology, Sector-26, Chandigarh, Punjab, India

I. Tyagi
Computer Science and Engineering Department, Amity University, Uttar Pradesh, India
e-mail: ishita27t@gmail.com

S. Bhalla et al. (eds.), *Intelligent Computing and Information and Communication*,
Advances in Intelligent Systems and Computing 673,
https://doi.org/10.1007/978-981-10-7245-1_71

1 Introduction

Main goal of web service is to communicate and exchange information by using a common platform among different applications built in different programming languages and operating system. The confidentiality and integrity of a web service need to be maintained as attacks on web application have increased.

The term software engineering is defined as a branch of engineering which is related to software development and management with the help of properly defined methods, principles and steps. The aim to provide the customer a reliable and efficient product. The software engineering contains various models for the software development. The models are used according to the requirements and other parameters required for software development and management. Since, one of the most important phase of any model is testing phase, so the testing should be done properly and carefully to get reliable results. It is not practically possible to implement exhaustive testing. There can be both valid and invalid inputs because it is very difficult to perform testing on all the test cases available. There are many reasons like time constraints, budget etc. because of which exhaustive testing is not possible. It is not easy to completely test because of the design problems. In regression testing, testing is done on the changes made in existing software to check whether the existing software is working properly or not after the changes has been done. Therefore, retesting is performed to detect the new faults found. This type of testing is performed again and again after the changes has been made in the pre-existing software. Changes can be of different types such as adding extra features, changes in the configuration etc. [1]. But this testing increases the budget of the software or the product because of the features enhanced and also a lot of maintenance is required.

Various methods are used for test case reduction and optimization for a web service. But the technique should be selected in such a way that budget must be less. Regression testing also creates a large number of test suites which consumes a lot of time in testing and many other problems are faced [2, 3]. The test case reduction and prioritization need to be achieved depending on various parameters such as branch coverage and also on basis of fault coverage etc. so, [4]. The technique used to analyze the web service based on branch or code coverage and also the fault detection for reduction of test case and prioritisation is bacteriologic algorithm (BA). The test cases generated and also other requirements are mapped with the branch coverage and fault coverage of the code of the web service. This helps in calculating fitness of the code of the web service based on branch coverage and fault detection capability.

2 Terminologies

There are some terms that are explained under this section which will be used in the further sections of the paper.

2.1 Test Case

Test case is a collection the preconditions, post-conditions, test data and results which are documented for a specific test scenario so that compliance can be verified against the requirement [3]. For execution of the test, test case is the starting point. Some of the parameters of test case are: Test case ID, Test data, Expected results, Actual results etc.

2.2 Test Suite

A group of test cases is known as test suite. The test execution status can be reported by the testers with the help of test suites. Each test case can used in one or more test suites. Number of test cases are there in a test suite. Before creating test suites, a test plan is made. According to the scope and the cycle test suites are generated. There can be number of tests viz-functional or Non-Functional.

2.3 Test Case Minimization

It is defined as process to make lesser number of test cases with the help of making test suites from test cases. It is used for reducing the cost of resources, time taken in execution etc. The test suites should be created in such a way that satisfy all the requirements. The main aim of test suite is remove redundancy that is created because of the test cases. There are different techniques used for removing redundancy such as genetic algorithm etc.

2.4 Test Case Prioritization

There are number of test suites available in regression testing [2]. Since, it is not easy because of the time constraints to test all the test suites over the code, we use test case prioritization. It can be defined as ordering the test suites that follow one or more criteria. The aim is to find the faults by selecting least no of test cases. It is

used so that the best results are found at the time of the result. There can be many be reasons for prioritizing the test suites: fault detection rate should be increased. Code coverage should me more and at a faster rate. Every system has a different architecture, Therefore, prioritization can also be done on basis depending on the system architecture. There are some parts of the architecture that have a major effect on the entire system, they should be tested properly. The components or parts of system can be tested individually also for test case prioritization.

2.5 Exhaustive Testing

The testing in which the testing is done for the all the test cases that are possible keeping in mind the quality [5, 6]. Every possible case is made but this type of testing is impractical. A product is considered to be perfect if the exhaustive testing of a software is done. It is actually not easy to pass exhaustive testing. There are strict deadlines because of which this type of testing becomes impossible to use. Some of the inputs become invalid after particular time period. Therefore, time constraint is also an issue. It is impossible if real time scenarios need to be tested like temperature etc. Also, it is impractical to test all the combinations of each user or many users [7]. All the outputs also cannot be checked.

2.6 Web Service

Goal of web service is to communicate and exchange information by using a common platform among different applications built in different programming languages and operating system. A web service contains UDDI, SOAP and WSDL.

2.7 Genetic Algorithm

The optimization problem that are related to natural selection are solved using the algorithm i.e. genetic algorithm. It changes each solution of the population again and again. In each step, the children are generated for the upcoming production from the parent test cases or test suites. And, as the generation passes, best or favourable results are seen. This type of algorithm is used in solving the issues that cannot use optimization algorithms etc. For knowing the nature of chromosome, fitness is calculated. There are various steps applied in this algorithm:

1. First step is selection in which test suites are selected and then crossover and also mutation are used over the test suites. The chromosomes or test suites are chosen generally which are having greater fitness.

2. Second step is crossover in which the successors are combined and then form new children.
3. The next operator used is mutation in which the current descendants are changed automatically.

2.8 Bacteriologic Algorithm

Mala et al. [8] this algorithm is the alternative form of the GA algorithm. It has slightly different steps because it follows the bacteria's nature. There is no operator like crossover present in it. It shows the properties of a bacteria. It only uses the operator like selection and mutation from the above algorithm referred. It is different because it contains a new function i.e. "memorization function" which uses the bacteria which is the most fit in each population. There is basic diagram showing the flow of the entire process.

1. First step is selection in which test suites are selected and then crossover and also mutation are used over the test suites. The chromosomes or test suites are chosen generally which are having greater fitness.
2. The next operator used is mutation in which the current descendants are changed automatically.
3. Next step is to find the "fitness" of the code using its formula.
4. Then, best test suite is memorized out of each population for addition to the next step/generation.

3 Related Works

Krishnamoorthi and Mary [1] in their paper *"Regression Test Suite Prioritization using Genetic Algorithms"* said that there are ways to use prioritization method with the help of genetic algorithm. The parameters checked are the effectiveness and time consumption. The faults are found which helps in knowing how effective is the proposed method. Time coverage plays an important role here. It helps in knowing the problems faced during time-aware prioritization. The paper also says also tells about the methods which help in reducing time above of prioritization.

Singh et al. [9] in their paper *"A Hybrid Approach for Regression Testing in Interprocedural Program"* mentioned that regression testing is performed again and again after the changes has been made in the pre-existing software. Changes can be of different types such as adding extra features, changes in the configuration etc. Paper contains a new algorithm related to variable which is applied on the variables with the help of hybrid method. The paper gives an algorithm for regression testing. Various codes have been analyzed to find a conclusion. The accuracy and performance according to the conclusion is much higher.

Jatain and Sharma [2] in their paper *"A Systematic Review of Techniques for Test Case Prioritization"* explained that there are many important activities that a maintenance phase faces like deletion, correction etc. For the retesting of the software retesting is done as modifications are done in the software time to time. But it Is expensive to retest again and again. There are many methods present for regression testing. The tester can choose the method according to their need and can do testing. Here, various methods used by different researchers for prioritization are presented for regression testing. Also different algorithms for the prioritization process is used. It helps in knowing scope of different methods. Also, explains the difficulties faced in prioritization based on requirements.

Joshi [5] in his paper *"Review of Genetic Algorithm: An Optimization Technique"* told about a technique which can be used for the optimization of the whole. After this there are many methods that come under genetic algorithm are used for optimization. The paper explains the basic of the algorithm and gives the proper flow of the "GA algorithm". Also, the operators of the algorithm are reviewed properly. It tells about the origin of the algorithm. There are many other optimization technique other than GA that are explained in the paper. Also it tells about different techniques of using the operators of the given algorithm.

Garg and Mittal [6] *"Optimization by Genetic Algorithm"* in their paper said the optimization problem that are related to natural selection are solved using the algorithm i.e. genetic algorithm. This research paper explains about the algorithm and the technique of optimization. The optimization problem that are related to natural selection are solved using the algorithm i.e. genetic algorithm. It changes each solution of the population again and again. In each step, the children are generated for the upcoming production from the parent test cases or test suites. The optimization method is related or present in every field like engineering etc. This paper explains about the "dejong function" and the experiments have been performed using this function in genetic algorithm. The paper with the help of this function gives better results. There is also the use "unimodal and multimodal benchmark functions" for the optimization. The results are represented using graphs and tables.

Ramesh and Manivannan [7] *"Test Suite Generation using Genetic Algorithm and Evolutionary Techniques with Dynamically Evolving Test Cases"* in their paper explained a new way for making test oracles with the help of evolutionary algorithm. This gave positive results which helped in finding bugs in various classes. This method is also easy to use. Test oracles take a lot of time and is difficult to make meaningful test cases if done manually on the data generated. In this paper coverage parameter is used for optimization of test suites. Also, branch coverage parameter has been used.

Shahid and Ibrahim [10] *"A New Code Based Test Case Prioritization Technique"* in their papers said that test cases and test suites play an important role in software testing. Validation is done of the software or product that is under inspection with the help of test cases and suites. It becomes difficult to test all the cases for all the code. Therefore, prioritization is used which can help in improving the effectiveness and also help in saving time. Here, a new algorithm or approach has been used for prioritization method and is done on basis of code covered. The

algorithm is used on a case study for the results which are good and promising. The test case that cover most part are considered more important.

Mala et al. [8] *"A hybrid test optimization framework-coupling genetic algorithm with local search technique"* in their paper explained that there are three different algorithms that are used out of which "HGA algorithm" is used for making test cases of better quality. For this to happen path coverage and score of mutation is analyzed of every test case. The best test cases are selected having greater path coverage and score of mutation. It also takes less time as test cases get reduced. The test cases analyzed using this method are compared with the other algorithm on basis of efficiency. The other methods used are genetic and bacteriologic algorithm.

4 Architecture and Algorithm Used

4.1 Flow Diagram

The basic flow of the algorithm that is used to find the results is explained in Fig. 1.

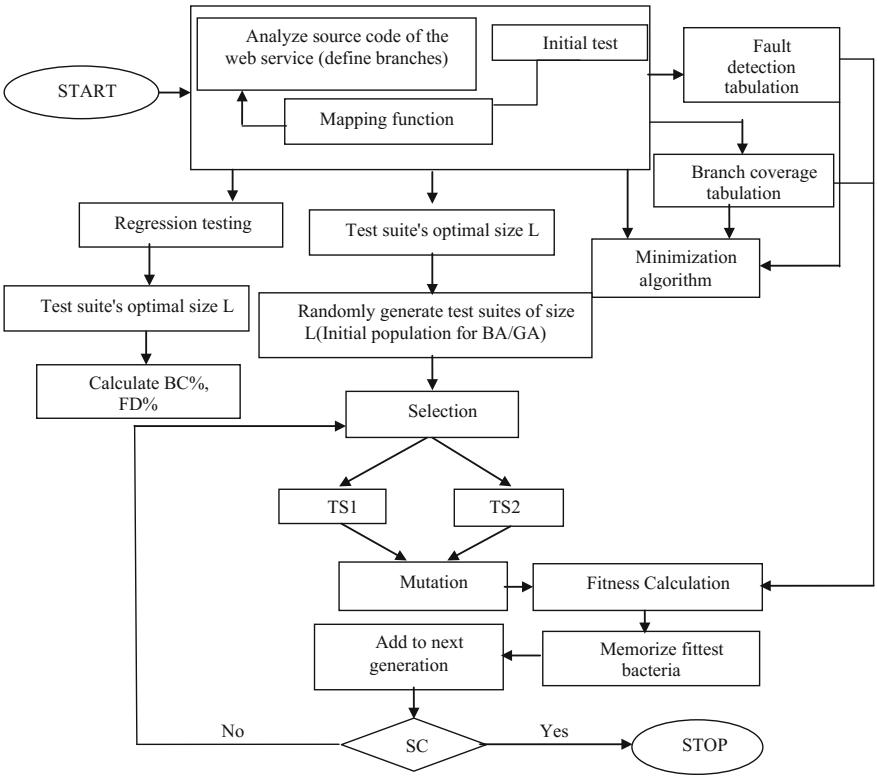

Fig. 1 Flow diagram

4.2 Algorithm Applied

Source code is the input i.e. *P*
There is a set/group of initial test cases (*given by the user*) *Q*
Reduced and prioritized test suites is the O/P.
Algorithm:

1. Find the branches in the code P which act as requirements from P.
   R_b = {B1, B2, B3 … B_n}
2. Initial test cases(Q) are mapped with R_b
   $BranchCoverage_{table}$ = MappingFunction (Q, R_b)
3. In code P, search use of the variables used.
   R_c = {z_1(line x, line y)… z_n(line x, line y)}
4. Q is mapped with R_c, $FaultDetectAbility_{table}$ = MappingFunction(Q, R_c)
5. Search for test suite (T) having maximum branch coverage and fault detection ability.
   T = Search($FaultDetectAbility_{table}$, $BranchCoverage_{table}$, Q)
6. Optimal length of test suites is to be searched,
   OL = SearchLength (T)
7. Make any random test suites ts_i = {ts_1, ts_2, ts_3 … ts_n} having length OL from Q
8. Use bacteriologic algorithm over ts_i

   a. Fitness calculation for every test suite.
      fitness = (weight1* branch coverage) + (weight 2* fault detection analysis)
      where branch coverage is obtained from Table 1.
      fda = fault detection ability (obtained from Table 2)
      w1 and w2 are weights ranging between 0 and 1.
   b. Arrange test suites present in tsi such that test suites with high fitness comes first.
   c. Apply mutation
   d. Fitness to be calculated again
   e. Learn the most fit test suite

Table 1 Value to be searched

Test cases	Search value
1	2
2	3
3	7
4	10
5	1
6	9
7	4

Table 2 Branch coverage

Test case	Search value	Branch b1	Branch b2	Branch b3	Branch b4
1	2	1	0	1	1
2	3	1	1	1	1
3	7	1	1	0	1
4	10	1	0	0	1
5	1	1	0	1	1
6	9	1	1	0	1
7	4	1	1	0	1

f. Most fit test suite is must be added to next level/generation
g. Repeat the steps from a and stop when stopping criteria is met. (stopping criteria here can be number of levels/generations)
h. Rearrange all memorised bacteria in descending order of fitness, to prioritize them.

4.3 Formula Required

1. "Branch Coverage" is,
   (no. of branch covered)/(total no. of branches)*100.
2. "Fault Detection analysis" is, ((Variable(p,q)covered)/(TotalVariable(p/q)covered))*100
3. "Fitness" is,
   (Weight1*BranchCoverage) + (Weight2*Fault Detection analysis) and, the value of the weight can be in between or 0 or 1.

5 Case Study

To get a more practical idea the code of the web service of binary search is observed and is passed as the input into the algorithm. The code is given below:

```
int * BinarySearch (int value)
{
unsigned int A = 0, B = array_length(array), S;
while (A<B)
{
S = (A+B− 1)/2;
if (value = = array [S])
return array+S;
else if (value<array [S])
B = S;
```

```
else
A = S+1;
}
return null;
}
```

The variables used in the above code i.e. *A*, *B* and *S* are taken as input for creating test cases from the user.

6 Results and Discussion

This part of paper shows the result. First step is to take input from the user to generate the test cases. The code is divided into four branches to know the branch coverage. Each line of the code is assigned a number to find the fault detection coverage. The first step is to input the array for binary search code and set the number of test cases to be generated. Then the test cases are formed and values to be searched are given as input by the user. For example:

Array given: {2,3,4,7,9}

The value to be searched are given in Table 1. For the above array, the branch coverage is calculated for the values given by the user in Table 1.

The branch coverage analysis is given Table 2 where, 1 represents that branch has been covered to search the particular number entered by the user and 0 represents that branch has not been covered to search the particular number entered by the user.

Table 3 represents the fault coverage where variable(p, q) is represented as, L (3,6) means L is the variable used in the code and (3,6) is the line in which it initialized and used respectively. Also, 1 represents that test case is covering line p and line q for that particular test case.

After the fault coverage analysis the regression testing is performed on the test cases generated by the user in Table 1. All the possible suits are generated in Table 4 using regression testing (Fig. 2).

Table 3 Fault coverage table

Test case	L (3,6)	R (3,6)	M (6,8)	M (6,10)	M (6,12)
1	1	1	1	0	1
2	1	1	1	1	1
3	1	1	1	0	1
4	1	1	0	0	1
5	1	1	0	1	1
6	1	1	1	0	1
7	1	1	1	0	0

Table 4 Table showing the results of fit calculation for all possible test suite using Regression testing

Test suite dD	Test case	Bc	Fda	Fit
1	1,2	100	100	100
2	1,3	100	100	100
3	1,4	75	80	80
4	1,5	75	80	80
5	1,6	100	100	100
6	1,7	100	100	100
7	2,3	100	100	100
8	2,4	100	100	100
9	2,5	100	100	100
10	2,6	100	100	100
11	2,7	100	100	100
12	3,4	75	80	80
13	3,5	100	100	100
14	3,6	75	80	80
15	3,7	75	80	80
16	4,5	75	80	80
17	4,6	75	80	80
18	4,7	75	80	80
19	5,6	100	100	100
20	5,7	100	100	100
21	6,7	75	80	80

Also, branch coverage, fda(fault detection analysis) and fitness is calculated for all the possible test suites of pair 2 in Table 4. The formula used for:

1. "branch coverage" is,

   (no. of branch covered)/(total no. of branches)*100.

2. "fault detection analysis" is,

   ((variable(p, q)covered)/(total variable(p/q) covered))*100

3. "fitness" is,

   (weight1* branch coverage) + (weight 2* fault detection analysis)

4. and, the value of the weight can be in between or 0 or 1.
5. For example. In Table 4:

- For test suite id 18 i.e. 4,7, 3 out of total 4 branches are being covered. Therefore, (3/4)*100 = 75%
  Similarly, for fault detection analysis, (4/5)*100 = 80%
  Therefore, after putting the values in formula of fitness, answer is 80%.

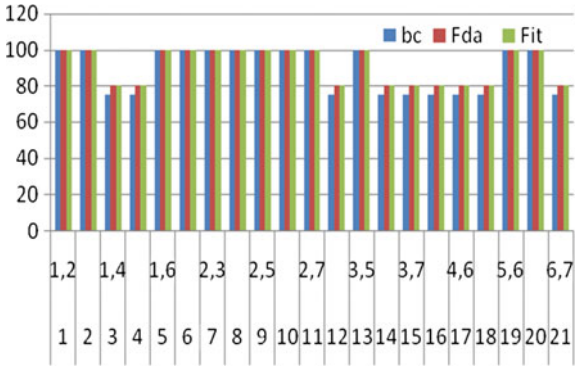

Fig. 2 Graph showing bc, fda and fit calculated for regression testing

- For test suite id 19 i.e. 5,6, 4 out of total 4 branches are being covered. Therefore, (4/4)*100 = 100%
  Similarly, for fault detection analysis, (5/5)*100 = 100%
  Therefore, after putting the values in formula of fitness, answer is 100%.

Next step is to calculate fitness using BA algorithm in which minimization algorithm is applied and best possible test suites are calculated After the application of minimization algorithm, the bacteriologic algorithm is used to find the fittest bacteria and reduces the number of test suites in Table 5 using mutation operator. The final answers are shown in Table 5; (Fig. 3).

Table 5 Table showing results after applying BA algorithm

Test suites	Test cases	Bc	Fda	Fit
1	5,9	100	100	100
2	5,7	100	100	100
3	2,9	100	100	100
4	2,7	100	100	100
5	6,7	100	100	100
6	6,3	100	100	100
7	2,7	100	100	100
8	2,3	100	100	100
9	8,6	100	100	100
10	8,2	100	100	100
11	8,9	100	100	100
12	2,5	100	100	100

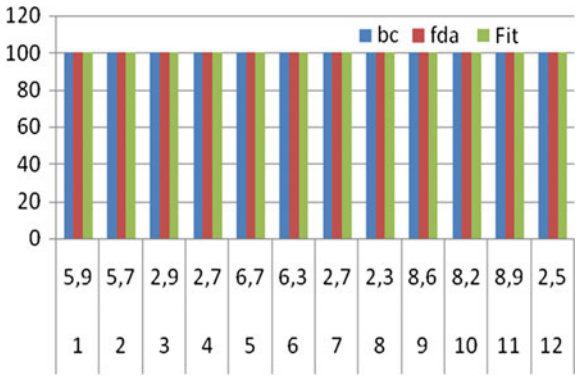

Fig. 3 Graph showing bc, fda, fit calculated using bacteriologic algorithm

7 Conclusions and Future Work

The paper is about the web service being developed and its code being tested for fitness calculation using regression testing. The aim is to reduce and order the test suites in such a way that all the best results are obtained rather than creating and testing number of test cases as done in regression testing. The criteria to analyze the code is branch coverage and fault detection analysis. Therefore, bacteriologic algorithm is implemented on the code of the web service so that improved results are obtained rather than using regression testing for fitness calculation of the code. The usage of this approach is shown with the help of the example and the graphs showing the results generated using regression testing and then using bacteriologic algorithm on the same piece of code to get reduced and ordered results. The result have been shown in the result section of the paper. The results show the reduced set of test suites having the best fitness. Also, the terms that are used in the paper are explained in the terminologies section. Further analysis can be done in future by comparing the results of the various web service's code using the same algorithm. Also, the results of one web service can be compared with many other algorithm derived in future to make improvements and get best result for test case reduction and prioritization using code coverage and fault detection as parameters.

References

1. Krishnamoorthi, R. and Mary, S.A.S.A, *"Regression test suite prioritization using genetic algorithms"*, International Journal of Hybrid Information Technology, Vol. 2, No. 3, pp. 35–51, 2009.
2. Aman Jatain, Garima Sharma, *"A Systematic Review of Techniques for Test Case Prioritization"*, International Journal of Computer Applications (0975–8887), Volume 68, No. 2, April 2013.
3. Presitha Aarthi. M, Nandini. V, *"A Survey on Test Case Selection and Prioritization"*, International Journal of Advanced Research in Computer Science and Software Engineering, ISSN: 2277 128X, Volume 5, Issue 1, January 2015.

4. R. Beena, Dr. S. Sarala, *"Code coverage based test case selection and prioritization"*, International Journal of Software Engineering & Applications (IJSEA), Vol. 4, No. 6, November 2013.
5. Gopesh Joshi, *"Review of Genetic Algorithm: An Optimization Technique"*, Volume 4, Issue 4, April 2014.
6. Richa Garg, Saurabh mittal, *"Optimization by Genetic Algorithm"*, International Journal of Advanced Research in Computer Science and Software Engineering, ISSN: 2277 128X, Volume 4, Issue 4, April 2014.
7. K. Ramesh and P. Manivannan, *"Test Suite Generation using Genetic Algorithm and Evolutionary Techniques with Dynamically Evolving Test Cases"*, International Journal of Innovation and Scientific Research, ISSN 2351-8014 Vol. 2 No. 2 Jun. 2014.
8. Dharmalingam Jeya Mala, Elizabeth Ruby, Vasudev Mohan(2010), *"A Hybrid Test Optimization Framework-Coupling Genetic algorithm with local search technique"*, Computing and Informatics, Vol. 29, 2010.
9. Yogesh Singh, Arvinder Kaur and Bharti Suri, *"A Hybrid Approach for Regression Testing in Interprocedural Program"*, Journal of Information Processing Systems, Vol. 6, No. 1, 2010.
10. Muhammad Shahid and Suhaimi Ibrahim, *"A New Code Based Test Case Prioritization Technique"*, International Journal of Software Engineering and Its Applications, Vol. 8, No. 6, 2014.