A Study of performance analysis of Priotization Techniques through Regression

Jyothi[1], Dr.Jeetendra Sheethlani[2]
[1](Computer Science Department, SSUTMS, INDIA)
[2](Computer Science Department, SSUTMS, INDIA)

Abstract: Test case prioritization involves scheduling test cases in an order that increases the effectiveness in achieving some performance goals. One of the most important performance goals is the rate of fault detection. Test cases should run in an order that increases the possibility of fault detection and also that detects the most severe faults at the earliest in its testing life cycle. In this paper, we develop and validate requirement based system level test case prioritization scheme to reveal more severe faults at an earlier stage and to improve customer-perceived software quality using Genetic Algorithm (GA). For this, we propose a set of prioritization factors to design the proposed system. In our proposed technique, we refer to these factors as Prioritization Factors (PF). These factors may be concrete, such as test case length, code coverage, data flow, and fault proneness, or abstract, such as perceived code complexity and severity of faults, which prioritizes the system test cases based on the six factors: customer priority, changes in requirement, implementation complexity, completeness, traceability and fault impact. The goodness of these orderings was measured using an evaluation metric called

Keywords: Test Case Prioritization, Regression Testing, Genetic Algorithm (GA), Prioritization Factors, Requirement Factor Value (RFV), Test Case Weight (TCW)

I. INTRODUCTION

The introduction of the paper should explain the nature of the problem, previous work, purpose, and the contribution of the paper. The contents of each section may be provided to understand easily about the paper.

Regression testing, which intends to ensure that a software program works as specified after changes have been made to it, is an important phase in software development lifecycle [12]. Regression testing is the re-execution of some subset of test that has already been conducted. In regression testing as integration testing proceeds, number of regression tests increases and it is impractical and inefficient to re execute every test for every program function if once change occurs. It is an expensive testing process used to detect regression faults [10]. Regression testing has been used to support software testing activities and assure acquiring an appropriate quality through several versions of a software product during its development and maintenance [1]. Regression testing is an important and yet time consuming software development activity. It executes an existing test suite on a changed program to assure that the program is not adversely affected by unintended amendments. Test suites can be large and conducting regression tests is tedious [6]. Regression testing assures the quality of modified service-oriented business applications against unintended changes. The test case prioritization is important in regression testing. It schedules the test cases in a regression test suite with a view to maximizing certain objectives which help reduce the time and cost required to maintain service-oriented business applications. Existing regression testing techniques for such applications focus on testing individual services or workflow programs [7]. Test case prioritization seeks to find an efficient ordering of test case execution for regression testing. The most ideal ordering of test case execution is one that reveals faults earliest. Since the nature and location of actual faults are generally not known in advance, test case prioritization techniques have to rely on available surrogates for prioritization criteria [5]. Test suite prioritization is a regression testing technique where test cases are ordered such that faults can be detected early in the test execution cycle. This is useful because tests accumulate over multiple revisions and versions of the system and it is not feasible to execute all the tests in a limited amount of time [23]. Test case prioritization is used in regression testing, at the test suite level, with the goal of detecting faults as early as possible in the regression testing process, given a test suite inherited from previous versions of the system. There are many techniques for prioritizing test cases based on various forms of information such as code coverage or modification history [2]. In test case prioritization techniques two dimensions are considered. The first is granularity and the second dimension is the prioritization strategy [8]. Over the lifetime of a large software product, the number of test cases could drastically increase as new versions of software are released. Because the cost of repeatedly retesting all test cases may be too high, software testers tend to remove redundant or trivial test cases to construct a reduced test suite for regression testing at a reasonable cost [22]. After development and release, software undergo regress maintenance phase of ten to fifteen years. Modifications in software may be due to change in customer's requirements or change in technology or platform. This leads to release of numerous versions or editions of the existing software. Also in case of the version or edition's test only modified and affected parts are to be tested to impart confidence in the modified software which is the process of regression testing [20]. Test case prioritization is an aspect of regression testing that permutes a test suite with the view of maximizing a test goal. Test case prioritization does not discard any test cases, having the advantage of not impairing the fault detection ability of the test suite as a whole [13]. The main disadvantage in regression testing is that it is an expensive process used to validate modified software [18]. The test case prioritization in regression testing can be used in various system such as in Time Aware Test Case Prioritization where experiments are performed on two subject programs involving four techniques two techniques for an approach to time aware test case prioritization based on genetic algorithms, and four traditional techniques for test case prioritization using integer linear programming [4]. Web Applications Testing where several test suite prioritization strategies for web applications are performed and these strategies improved the rate of fault detection for web applications and their preexisting

test suites [3]. For Audit Testing of Evolving Web Services where an approach to the prioritization of audits test cases using information retrieval is performed.
This approach matches a service change description with the code portions exercised by the relevant test cases [19]. For Audit Testing of Web service where a test case prioritization method specifically tailored for audit testing of services is performed. The method is based on the idea that the most important test cases are those that have the highest sensitivity to changes injected into the service responses [24]. In Black-Box Testing Environment an approach to test suite reduction for regression testing in black box environment has been proposed. The reduced regression test suite has the same bug finding capability and covers the same functionality as the original regression test suite [28]. In Recent Priority Algorithm where a model that achieves 100% code coverage optimally during version specific regression testing is done. Here prioritization of test cases is done on the basis of priority value of the modified lines covered by the test case [14].

II. A SURVEY OF RECENT RESEARCH IN THE FIELD

A handful of researches have been presented in the literature for the prioritization of regression testing test cases. Recently, utilizing artificial intelligence techniques like Greedy Algorithm and Genetic Algorithm (GA), in Prioritization has received a great deal of attention among researchers. A brief review of some recent researches is presented here. Yogesh et al. [9] have proposed an approach that variables were vital source of changes in the program and test cases should be prioritized according to the variables of any changed statement and variables computed from the variables of changed statements. In support of their prioritization approach they extended to validate the effectiveness of prioritized test cases with respect to data flow technique. The experimental study investigating the effectiveness of their prioritization approach by considering programs. The results obtained were encouraging and support their work to validate the prioritization technique with respect to DU or DC paths of data flow testing technique. Here the variable based prioritization of test cases for regression testing is in conformance with the data flow testing strategy. R.Kavitha et al. [15] have proposed a prioritization technique to improve the rate of fault detection of severe faults for Regression testing. Here, two factors rate of fault detection and fault impact for prioritizing test cases are proposed. The proposed algorithm was validated by analyzing two sets of industrial projects. Test case prioritization techniques schedule test cases for execution so that those with higher priority, according to some criterion are executed earlier than those with lower priority to meet some performance goal. Results indicate that the proposed technique lead to improved rate of detection of severe faults in comparison to random ordering of test cases. And also it was tested experimentally that the number of test cases runs to find the entire fault was less in case of proposed prioritization technique. The results prove that the proposed prioritization technique was effective. Arup et al. [16] have proposed a method to prioritize the test cases for testing component dependency in a Component Based Software Development (CBSD) environment using Greedy Approach. An Object Interaction Graph (OIG) was being generated from the UML sequence diagrams for interdependent components. The OIG was traversed to calculate the total number of inter component object interactions and intra component object interactions. Depending upon the number of interactions the objective function was calculated and the test cases were ordered accordingly. This technique was applied to components developed in Java for a software system and found to be very effective in early fault detection as compared to non-prioritize approach.

III. TEST CASE PRIORITAZATION IN REGRESSION TESTING

As the name suggests, regression testing involves saving and reusing test suites which have been created for earlier version releases of the software. By reusing these already created test cases, the costs of the designing and creating of test cases can be amortized across the lifetime of a system. Empirical studies have allowed researchers to compare regression testing techniques earlier; however these studies suffer from severe limitations in their abilities to assess cost-benefit tradeoffs relative to practical testing situations. The main limitations include: Context factors, Lifetime factors and Costbenefit models. These limitations make it difficult to empirically compare the various regression testing techniques or when evaluated could lead to improper assessment of costs and benefits in practical situations. This ultimately leads to inaccurate conclusions about the cost-effectiveness of techniques and decisions of engineers relying on such conclusions to select techniques for regression testing.

Regression testing is the process of validating modifications introduced in a system during software maintenance. Regression testing is an expensive process used to validate modified software. As the test suite size is very large, system retesting consumes large amount of time and computing resources. This issue of retesting of software systems can be handled using a good test case prioritization technique. A prioritization technique schedules the test cases for execution so that the test cases with higher priority executed before lower priority.

An improved rate of fault detection during regression testing can let software engineers begin their debugging activities earlier than might otherwise be possible, speeding the release of the software. An improved rate of fault detection can also provide faster feedback on the system under test and provide earlier evidence when quality goals have not been met, thus allowing strategic decisions about release schedules to be made earlier than might otherwise be possible. Test case prioritization techniques improve the cost-effectiveness of regression testing by ordering test cases such that those that are more important are run earlier in the testing process. Prioritization can provide earlier feedback to testers and management, and allow engineers to begin debugging earlier. It can also increase the probability that if testing ends prematurely, important test cases have been run.

Prioritization Factors

Computation of certain factors such as (1) customer assigned priority of requirements, (2) implementation complexity, (3) changes in requirements, (4) fault impact of requirements, (5) completeness (6) traceability and (7) Execution time, is essential for prioritizing the test cases because they are used in the prioritization algorithm. According to these factors weights are defined for each test case in the software. Based on prioritization the evaluation cost and time could be decreased by focusing only on particular test cases.

Customer-Assigned Priority (CP)

CP is the measurement of importance to the customer's need. Customer's need vary from 1 to 10 and assigned by customer itself, where 10 is used to identify the highest customer priority. It is essential to improve the customer perceived value for the development of the customer. Software functions never, infrequently used 45%, occasionally used 19% and always used 36%, all are approximate idea. Maximum effort should be utilized for the faults identification which takes place on regular interval otherwise these faults results in continuous failures. It was proved that customer-perceived value and satisfaction can be improved by focusing on customers want and development.

Implementation Complexity (IC)

Implementation complexity is the subjective measure of the complexity anticipated by the development team in implementing the need and it is evaluated initially. Value from 0 to 10 assigned by the developer on the basis of its application complexity and by using larger value higher complexity is denoted. No of faults increases as the requirement is become high in implementation complexity.

Changes in Requirements (RC)

Developer assigned a measure and the range varies from 1 to 10 which used to indicate the number of times a need is changed in the development cycle by taking its origin date as a reference is called RC. Need changes to 10 times if the volatility values for all the needs are represents on 10 point scale. . The number of changes for any requirement i divided to the highest number of changes for any requirement among all the project requirements yields the change in requirement i R of that requirement i . If the ith requirement is changed M times and N is the maximum number of requirements then the requirement change of I , i R can be calculated as

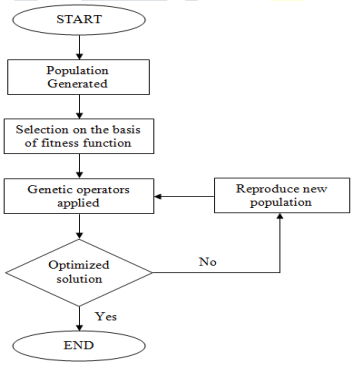
$$iR = M \div N \times (1)$$

Approximately, 50% of error introduces in the requirement phase of all faults detected in the project. The most important factor for the failure of the project is the change in the requirement phase.

Fault Impact of Requirements (FI)

FI allows the development team to identify the requirement which has had customer reported failures. As a system evolves to various versions, the developer can use the previous data which is collected from versions to identify requirements that are likely to be error prone. Fault impact is based on the number of field failure and in-house failures and it is consider to those requirements which have already been launched product. In this work, we propose to calculate fault impact of a requirement, based on the severity of the fault identified in the previous run. Fault severity 1, 2, 3, 4 and 5 correspond to the severity values 25, 24, 23, 22 and 21. If a requirement i with the set of t test cases, discovered the set of d faults of the set of severity V , then the severity i S of the requirement i is computed as and if { 1.... } i
S = S ∀ i = n where n is the total number of requirements, is the set of all severities of each requirement i , then the fault impact i FI of a requirement i is computed as Test efficiency can be improved by focusing on the function that is likely to contain higher number of faults.

Flow Diagram:



IV. CONCLUSION

In this paper the regression testing based test suite prioritization technique is illustrated. A new prioritization technique is proposed for requirement based System level test cases to improve the rate of fault detection of severe faults. This paper recognizes and assesses the challenges coupled with regression testing test case prioritization. The proposed method uses most efficient factors to prioritize the test cases. The beneficiary thing in the proposed method is the trace events process. This factor identifies the important test cases in the project. The effectiveness of the proposed prioritization technique can be evaluated by using the APFD metric. In the proposed method, two types of application projects are used. The proposed method gave the better rate for severe faults detection through results. Also it is tested experimentally that the numbers of test cases run to find the injected fault is less in case of proposed prioritized execution of test cases. Based on the performance measure obtained, the proposed method is effectively prioritizing the test cases in both projects based on the factors and weightage assigned. This will reduce the cost of time of executing the entire project.

REFERENCES

[1]. Hyuncheol Park, Hoyeon Ryu and Jongmoon Baik, "Historical Value-Based Approach for Cost-cognizant Test Case Prioritization to Improve the Effectiveness of Regression Testing," In.Proc.of 2nd IEEE International Conference on Secure System Integration and Reliability Improvement SSIRI, Yokohama, pp.39-46, 2015.

[2]. Xiao Qu, Myra B. Cohen and Gregg Rothermel, "Configuration-Aware Regression Testing: An Empirical Study of Sampling and Prioritization," In.Proc.of the 2016 international symposium on Software testing and analysis, pp.75-85, 2016.

[3]. Sreedevi Sampathy, Renee C. Brycez, Gokulanand Viswanathy, Vani Kandimallaz and A. Gunes¸ Koruy, "Prioritizing User-session-based Test Cases for Web Applications Testing," In.Proc.of the 2014 International Conference on Software Testing, Verification, and Validation, Lillehammer, pp.141 - 150, 2014.

[4]. Lu Zhang, Shan-Shan Hou, Chao Guo, Tao Xie and Hong Mei, "Time-Aware Test-Case Prioritization using Integer Linear Programming," In.Proc.of the eighteenth international symposium on Software testing and analysis, 2016.

[5]. Shin Yoo, Mark Harman, Paolo Tonella and Angelo Susi, "Clustering Test Cases to Achieve Effective & Scalable Prioritisation Incorporating Expert Knowledge," In.Proc.of the eighteenth international symposium on Software testing and analysis, pp. 201 - 211, 2016.

[6]. Vengatesan K., and S.Selvarajan " Improved T-Cluster Scheme for combination gene scale expression data" International Conferenceon Radar, Communication and Computing (ICRCC), pp. 131-136. IEEE(2012)

[7]. Kalaivanan M., and K.Vengatesan" Recommendation system based on statistical analysis of ranking from user. International Conference on Information Comunication and Embedded Systems (ICICES), pp.479-484, IEEE, (2013)

[8]. K.Vengatesan, S.Selvarajan: The performance Analysis of Microarray Data using Occurance Clustering. International Journal of Mathematical Science and Engineering, Vol.3(2).pp69-75 (2014)

[9]. K Vengatesan, V.Karuppuchamy, S.Pragadeeswaran, A.Selvaraj, "FAST Clustering Algorithm for Maximising the Feature Selection in High Dimentional Data", Volume – 4, Issue – 2, International Journal of Mathematical Sciences and Engineering (IJMSE), December 2015.