Test Case Prioritization: A Review

Monika
CSE Department, DeenBandhuChhotu Ram University of Science and Technology,
Murthal, Haryana, India

Ajmer Singh
CSE Department, DeenBandhuChhotu Ram University of Science and Technology,
Murthal, Haryana, India

***Abstract*- Testing is a technique to certify the quality of developed software. When some modifications are being done in software regression testing is used to revalidate the software. The main motive behind this review is to introduce the enhancement in regression test case prioritization.**

***Keywords*- test case prioritization, metaheuristic algorithms, cost modal, time constraint.**

I. INTRODUCTION

Considering the software development and maintenance process one of the most critical activities is regression testing. Test suites which are developed by software developers for their software have being saved by them. These test suites are then used in the form of regression testing. Reuse of test suite and other activities related to regression testing have one half of the cost of software maintenance. Regression testing is very expensive but ensures that the software program will work according to its specification after changes have been made to it. Not only the cost these activities can consume an inordinate amount of time.

One of the approaches to overcome these problems testers runs those test cases first which have highest priority according to some criterion. This approach of regression testing is called test case prioritization.

Various approaches of regression techniques are [1]:-

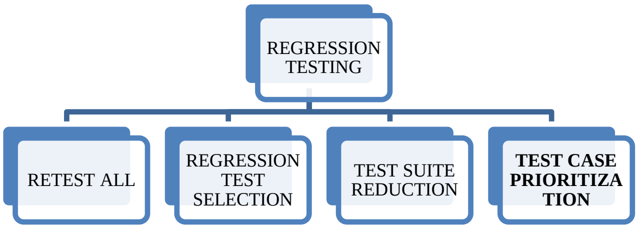

Fig 1 Approaches of regression techniques

1) Retest All
2) Regression Test Selection
3) Test Suite Reduction
4) Test Case Prioritization

Retest All - It is the most straightforward approach of regression testing. In this simply execute all the existing test cases in the test suite.
Regression Test Selection – It deal with the problem of selecting a subset of test cases from the test suite. These selected test cases are then used to test the changed parts of software.
Test Suite Reduction – This process has two parts. First is to identify the absolute or redundant test cases and second to eliminate those test cases.
Test Case Prioritization – Finally test case prioritization, it concerns with the recognition of idyllic ordering of test cases. That ordering should maximises desirable properties like early fault detection and number of fault detection. Another desirable property may be to minimise the testing cost.

II. Prioritization Techniques

Classification based on the characteristics of the prioritization algorithms-

A. Based on customer requirements
B. Based on coverage
C. Based on cost effective
D. Based on chronographic history

Based on customer requirements-

In these techniques various customer requirement factor are considered. Assign some weight to these factors. Test cases having high weight value are executed first and test cases with low weight value are executed later.
Hema Srikanth and Laurie Williams [2] present a technique in which they consider 3 factors

1) Customer-assigned priority on requirements (CP)
2) Requirement complexity (RC)

3) Requirement volatility (RV)

CP value is allocated by the customer. RC value is allocated by developer. Value of RV depends upon the number of times modifications are being done in the requirements.

$$WP = \sum(PF\ value * PF\ weight) \qquad (1)$$

WP = weighted prioritization
PF value is values assigned to prioritization factors. PF weight is weight assigned to prioritization factors.
Test cases are being ordered according to WP values .Test cases having the higher values are executed first.
R.kavitha et al. [3] also proposed a technique in which they consider 4 factors

1) Priority of requirements assigned by customer
2) Code implementation complexity assigned by developer
3) Changes in requirements
4) Fault impact

Equation used

$$RFVi = \sum_{j=1}^{3} Factor\ value\ j\ /3 \qquad (2)$$

Where 'n' requirements, 'j' factor value. RFVi is requirement factor value for each requirement 'i'.

Based on requirement factor values TCW test case weight is calculated according to equation

$$TCW = \left[\frac{\sum_{x=1}^{i} RFVx}{\sum_{y=1}^{n} RFVy}\right] * i/n \qquad (3)$$

Test cases are ordered according to values of TCW.

Ashraf et al. [4] present a requirement based technique in which they consider 6 factors

1) Customer priority
2) Requirement volatility
3) Requirement traceability
4) Implementation complexity
5) Execution time
6) Fault impact of requirement

They present a value based prioritization algorithm. Their algorithm works at 2 levels i.e. requirement level and testing level.
To get the net values calculations are being done on the values get from the above 2 levels. These values are further used for ordering of test cases. They also compare their algorithm with random prioritization algorithm. Comparisons show that their algorithm is more effective for early fault detection.

Based on coverage

For detecting faults earlier in testing, we have to achieve more coverage. These techniques test internal structure of data and may be consider as white box testing.
Wong et al. [5] propose a technique in which their criterion of test case prioritization is of increasing cost per additional coverage. They use the tool called ATAC an automatic testing tool for analysis in c for test case selection and minimization or prioritization.
Rothermal et al. [6] propose 4 coverage based techniques they are total coverage, additional coverage, statement coverage, branch coverage. For evaluation they used the Aristotle a program analysis tool. APFD is used for measuring the results. Their results shows that total coverage surpass the additional coverage prioritization.
Elbaum et al. [7] propose the version specific prioritization technique. They present 8 function level techniques they are

1) Total function
2) Additional function
3) Total FEP function
4) Additional FEP function
5) Total FI function
6) Additional FI function
7) Total FEP FI functional
8) Additional FEO FI functional

Total function and additional function are based on coverage. 4 statement level techniques also presented but in context of version specific. A comparison between function and statement level was being done in terms of rate of fault detection. APFD metric is used for fault detection. ANOVA and Bonferroni analyses were performed on all techniques. Optimal ordering is superior to all other techniques. Random is the worst one. In terms of the cost effectiveness function level are far better than statement level techniques.
Amitabh srivastava and Jay thigarajan [8] proposed a prioritization technique based on binary code. They gave a system called ECHELON.

- ECHELON prioritize the test cases based on the modification are being done in the program.
- ECHELON is an integrated part of Microsoft software development process.
- It uses simple and fast algorithm.
- It generates results within a few seconds thus saving time and resources.

Do et al. [9] present a controlled experiment. JUnit framework is a framework that is used by software developers for generating test cases for programs that are being implemented in java. JUnit framework helps the developers to write the test cases and to rerun these test cases whenever some modifications are being done in the program. There experiment is for finding the effectiveness of test case prioritization under this JUnit framework.
They present 6 block and method level techniques

1) Total block coverage
2) Additional block coverage
3) Total method coverage
4) Additional method coverage
5) Total DIFF method
6) Additional DIFF method

Three type of information are being used. They are coverage information, modification information and use of feedback. Modification information is for DIFF method. These techniques are specific to JUnit environment and a comparison was being done with techniques specific to C language [10].

The result shows that there is no effect of level of granularity and the modification information on the prioritization. Because the instrumentation granularity of java is different from C, so statement level techniques based on java were found better than function level techniques based on java.

Renee C. Bryce and Atif M. Memon [11] prioritizes test cases for interaction coverage. Mainly their work is for event driven software. They prioritize test cases based on five criterions

1) Unique event coverage-prioritize test cases so that they cover unique events as soon as possible.
2) Event interaction coverage- 2-way interaction and 3-way interaction.
3) Longest to shortest with respect to length of test cases.
4) Shortest to longest with respect to length of test cases.
5) Random test ordering-test cases are ordered randomly without any rule.

The result shows that if we want to achieve fastest fault detection rate our test suite must have leading percentage of 2-way and 3-way interaction.

Belli et al. [12] proposed techniques in this ordering of relevant events are being done. The events have many features. Events are prioritizing according to the importance of their features. Graph modal based approach is used for prioritization. Fuzzy c-Mean clustering algorithm is used for erection of events. No need of prior information in this approach. Run time complexity is $o(n^2)$. Where n is the number of events.

Do et al. [13] proposed a technique in which they want to find out what are the effect on a specific prioritization technique of variation in time constraint and also the effect on cost benefits of regression testing. They propose four techniques two are related to total and additional coverage and two are related to Bayesian network. The equation used in the technique is-

$$COST = PS * \sum_{i=2}^{n}(CS(i) + CO_i\,(i) + CO_r(i) + b(i) * CV_i(i) + c(i) * CF(i)) \qquad (4)$$

Additional techniques are found to be better than total. Result shows that time constraint play a noteworthy role in test case prioritization techniques.

Jiang et al. [14] proposed a Adaptive Random Test Case Prioritization Technique (ART). They propose nine new coverage based ART techniques. They are classified into three groups' maxmin, maxavg and maxmax. Their coverage is at statement level, function level and branch level.

1) ART-st-maxmin
2) ART-st-maxavg
3) ART-st-maxmax
4) ART-fn-maxmin
5) ART-fn-maxavg
6) ART-fn-maxmax
7) ART-br-maxmin
8) ART-br-maxavg
9) ART-br-maxmax

A comparison was being done between these techniques and random ordering, and these found 40 to 50% more effectual than random ordering. ART-br-maxmax is best among the group. In terms of revealing failure these are more efficient and statically more effectual than traditional coverage techniques.

Maia et al. [15] presents a metaheuristic algorithm called GRASP (greedy randomized adaptive search procedure). They do automatic test case prioritization with the help of GRASP. A metaheuristic algorithm finds out good solutions as well as optimal solutions. They compare the reactive GRASP approach with other search algorithms like greedy algorithm, additional greedy, genetic algorithm and simulated annealing. Their comparison is in terms of time performances and coverage. Their coverage criterions are block, decision and statement.

The results show that out of these five algorithms additional greedy is best one but reactive GRASP is not worse than that. GRASP surpassed the genetic algorithm, greedy algorithm and simulated annealing.

Dennis Jeffrey and Neelam Gupta [16] proposed a prioritization technique using the relevant slice. A program contains many statements. Some statements have no influence on the output produced by the test case but some statements have potential to influence the output produced the test. All these statement forms a group and this group correspond to relevant slice.

In this approach following factors are considered

- The number of statements in the relevant slice of the output of the test case.
- The number of statements that are not in the relevant slice of the output but are executed by the test case.

Equation used for calculating the weight of a test case is

$$TW = Reqslice + ReqExercise \qquad (5)$$

Reqslice is number of requirements presented in the relevant slice of output for the test case.

ReqExercise is number of requirements exercised by the test case.

Based on cost effective-

There are much kind of cost related to test cases like cost of analysis and cost of prioritization. In cost effective based techniques test cases are ordered for execution based on cost.

Leung and white [17] propose a cost modal that compare the various regression strategies. They divide the total cost into two parts

- Direct cost
- Indirect cost

Direct cost includes

1) System analysis cost C_a
2) Test selection cost C_s
3) Test execution cost C_e
4) Result analysis cost C_r

Indirect cost includes

1) Overhead cost
2) Tool development cost

One disadvantage of this technique is that they did not include the cost of undetected faults.

Alexey Malishevsky et al. [18] proposed cost modal of cost-benefit tradeoffs for regression testing. They did experiments for test case selection, test case reduction and test case prioritization and presents cost modals for them.

They used the cost factors like

- $C_a(T)$ cost of analysis
- $C_e(T)$ cost of execution
- $C_c(T)$ cost of result checking
- $C_s(T)$ cost of selection
- $C_m(T)$ cost of maintenance of the test suite

In experiment for test case prioritization they consider two factors cost of analysis and cost of prioritization $C_p(T)$. In their work they divide the testing process in two phase's one preliminary and second critical phase. These two phases have different costs. The result shows that the optimal ordering, total function coverage and additional function coverage have maximum saving.

Based on chronographic history-

In these type of prioritization techniques test execution history considered to be the main factor for prioritization of test cases.

Jung-Min-Kim and Adam Porter [19] proposed a history based test case prioritization technique. It is for regression testing in resource constrained environment. Their main motive behind this is to show that historical information can be useful for decreasing the cost and it may be useful in increasing the effectiveness of testing process.

They do comparison of some prioritization methods like LRU, random, safe random. LRU requires less total effort than random both in terms of median and average. Safe random requires less total effort than random in terms of median and average. Safe random is little bit better than LRU in terms of median and average.

Weakness of their cost modal is that they only take the consequence of last execution of the test case to calculate the selection probability of the test cases.

Park et al. [20] propose an approach that uses the historical information for cost-cognizant test case prioritization. This information improves the effectiveness of regression testing.

Factors consider are-

1) Function level gratuity
2) Historical information of the cost of the test cases
3) Fault severities of detected defects in a test suite
4) These factors are used to calculate the historical value of the test case and that value is used for test case prioritization.

A comparison is being done between their technique and functional coverage technique. Results show that in terms of APFD it better than functional level technique.

Fazlalizadeh et al. [21] make some changes in the technique of Kim and porter. If resource and time constraint environment is considered they motive is to give faster fault detection.

Factor consider are-

1) Historical effectiveness of the test cases
2) Execution history of the test cases
3) Last priority assigned to the test cases.

A comparison was being done with the random ordering. The box plots shows that it has faster fault detection and stability.

III. CONCLUSION

This paper present a review on regression test prioritization techniques which evaluates the research work related to the area. This paper summarises the research papers along with the techniques they compared. Explanation makes the researches understand the scope of working various techniques. We can conclude that there are so many techniques that are used for test case prioritization. Each technique has its advantages and disadvantages. According to requirement a tester can use any technique.

IV. ACKNOWLEDGMENT

I would like to place on record my deep sense of gratitude of Mr. Ajmer Singh, Assistant Professor at Deenbandu Chhotu Ram University of Science and Technology, Murthal, Haryana, for their valuable time and useful suggestions which are responsible for the work produced here.

REFERENCES

[1] S.Yoo, M.Harman, “Regression testing Minimisation, Selection and Prioritization: A Survey” Wiley InterScience DOI: 10.1002/000, 2007

[2] Hema Srikanth, Laurie Williams, “Requirements-Based Test Case Prioritization”.

[3] R.Kavitha, V.R.Kavitha. N. Suresh Kumar “Requirement Based Test Case Prioritization” 978-7-4244-7770-8 IEEE, 2010.

[4] E. Ashraf, A. Rauf, and K. Mahmood, “Value Based Regression Test case Prioritisation”, Proceedings of the World Congress on Engineering and Computer Science 2012 Vol I WCECS 2012, October 24-26, 2012, San Francisco, USA.

[5] W.E. Wong, J.R.Horgan, S.London, and A.Aggarwal, “A study of effective regression testing in practice”, Proceedings of the Eighth International Symposium Software Reliability Engineering,

[6] G. Rothermel, R.Untch, C.Chu, and M.J.Harrold, “Test case prioritization: An empirical study”, Proceedings of International Conference Software Maintenance, pp. 179-188, Aug. 1999.

[7] S.Elbaum, A.Malishevsky and G. Rothermel, “Prioritizing test cases for regression testing”, Proceedings of the International Symposium on Software Testing and Analysis, pp. 102-112, Aug.2000.

[8] A.Srivastava, and J.Thiagarajan, “Effectively prioritizing tests in development environment”, Proceedings of the International Symposium on Software Testing and Analysis, pp.97-106, July 2002.

[9] H.Do, G.Rothermel and Kinner, “Empirical studies of test case prioritization in a Junit testing environment”, Proceeding of the International Symposium on software Reliability Engineering, pp.113-114, NOV 2004.

[10] S.Elbaum, A.G.Malishevsky, and G.Rothermel, "Test case prioritization: A family empirical studies", IEEE Transactions on Software Engineering, Vol. 28, No. 2, pp. 159-182, Feb.2002.

[11] R.C. Bryce, A.M. Menon, "Test Suite Prioritization by Interaction coverage", Proceedings of the workshop on domain specific approaches to software test automation (DOSTA), ACM, pp. 1-7, 2007.

[12] F.Belli, M.Eminov, N.Gokco, "Coverage-Oriented, Prioritization Testing A fuzzy Clustering Approach and Case Study". In: Bondavalli.A.Brasileiro, F.Rajsbaum, S.(eds.) LADC 2007, LNCS, Springer, Heidelberg, Vol. 4746, pp. 95-110, 2007.

[13] H. Do, S. Mirarab, L. Tahvildari, G. Rothermel, "An Empirical Study of the effect of time constraints on the cost benefits of regression testing" Proceedings of the 16th ACM SIGSOFT International Symposium on Foundations of software engineering,pp71-82,2008.

[14] B.Jiang, Z.Zhang, W.K.Chan, T.H.Tse, "Adaptive Random test case prioritization" In Proceedings of International Conference on Automated Software Engineering, pp:233-243, 2009.

[15] C. L. B. Maia, R. A. F. do Carmo, F. G. De Freitas, G. A. L. de campos and J. T. De Souza, "Automated test case prioritization with reactive GRASP," In Proceedings of Advances in Software Engineering, pp.1-18, 2010.

[16] Dennis Jeffrey and Neelam Gupta "Test case prioritization using relevant slices" Department of computer science The University of Arizona TUCSON, AZ85721

[17] Harton K. N. Leung and Lee White "A Cost Modal to compare Regression Test Strategies" CH3047-8|91|0000|0201, IEEE 1991.

[18] Alexey G. Malishevsky, Joseph R. Ruthruff, Gregg Rothermel, Sebastian Elbaum "Cost-cognizant Test Case Prioritization" Technical Report TRUNL-CSE-2006-0004, Department of Computer Science and Engineering, University of Nebraska – Lincoln, 2006.

[19] J.M.Kim, A. Porter, "A History-Based Test Prioritization Technique for Regression Testing in Resource Constrained Environment" , Proceedings of the 24th International Conference Software Engineering, pp.119-129, May.2002.

[20] H. Park, H.Ryu, J.Baik, "Historical value-based approach for cost-cognizant test case prioritization to improve the effectiveness of regression testing", Proceedings of second International Conference on Secure System Integration and reliability, Improvement, pp. 39-46, 2008.

[21] Y.Fazlalizadeh, A.Khalilian, H.A.Azgomi, S.Parsa, "Incorporating historical test case performance data and resource constraints into test case prioritization" , Lecture notes in Computer Science, Springer, Vol. 5668, pp. 43-57, 2009.