Indian Journal of Science and Technology, *Vol 10(10), DOI: 10.17485/ijst/2017/v10i10/99661, March 2017*

ISSN (Print) : 0974-6846
ISSN (Online) : 0974-5645

A Survey on Test Case Prioritization Techniques for Regression Testing

G. Pardha Sagar[1,2]* and P. V. R. D. Prasad[1]

[1]Department of Computer Science and Engineering, KL University, Green Fields, Vaddeswaram, Guntur – 522502, Andhra Pradesh, India; pvrdprasad@kluniversity.in
[2]Department of Computer Science and Engineering, RMD Sinhgad School of Engineering (RMDSSOE), Survey No. 111/1, Pune-Mumbai Bypass Highway, Warje, Pune – 411058, Maharashtra, India; parthsagar.rmdssoe@sinhgad.edu

Abstract

Objective: The main intent of this research is to provide prioritization of test cases in regression testing for various applications. For that, several test case prioritization techniques which are classified based on various parameters are investigated. **Methods/Statistical Analysis**: In this manuscript, a survey has been made on various test case prioritization techniques for regression testing. Several test case prioritization techniques are presented for regression testing of various applications. One of the suggested technique is Reinforcement Learning (RL) based Hidden Markov Model (HMM) method for prioritizing test cases during regression testing of Graphical User Interface (GUI) applications. Results: This survey comprehensively studies the issues in test case prioritization techniques for regression testing. The performance of different methods is compared with various parameters such as Average Percentage Faults Detected (APFD), effect size, statistical testing. The mean values of APFD for RL-Based HMM model method is 0.68, for accumulated Q-value method is 0.62 and for statement coverage method is 0.61. **Findings:** The major findings in this survey are that test case prioritization is still at its primitive stages and more research is required to make it applicable in today's world. It can be applied to all streams of computer engineering if made practically feasible. Currently test case prioritization is only applied where there is no consideration of cost like safety critical software. Many existing methods are surveyed in this work and the findings suggest that model based test case prioritization is best in all aspects. **Application/Improvement:** Test case prioritization can be applied to all kinds of software once it is cost effective and practically feasible but currently its application is limited to some software components only. **Conclusion:** This survey investigates several test case prioritization techniques and provides the idea for efficient methods for future work.

Keywords: APFD, Effect Size, Prioritization, Regression Testing, Statistical Testing, Test Case

1. Introduction

Testing and verifying software systems is becoming increasingly important with the growth in the size and complexity of modern software systems. Due to which, many test cases which were produced during development are reused in regression testing increasing the overall cost of testing. Test Case Prioritization (TCP) provides an efficient solution to decrease the costs of regression testing. An efficient test case prioritization technique plays a vital role in increasing efficiency of testing while keeping the cost of testing in check. There are various examples of test case prioritization techniques such as Random Ordering prioritization, Optimal Ordering prioritization, Total statement Coverage Prioritization, Additional Statement Coverage Prioritization, Total Function Coverage Prioritization[1–4].

Various test case prioritization techniques depending on different parameters such as code coverage, fault detection, function coverage are suggested in this survey. The performance of the numerous prioritization techniques is also analyzed.

The remainder of this survey is organized as follows: Section2 describes the various parameters considered

*Author for correspondence

to classify test case prioritization techniques. Section 3 presents the insight to various TCP techniques. Section 4 analyzes the comparison of various techniques. Section 5 gives the conclusion of the survey.

2. Elements for Classification of Prioritization Techniques

Each and every test case prioritization technique should address one or more test case prioritization goals. The goals are generally referred as the objectives of TCP techniques. To consider the prioritization in the regression testing some of the prioritization goals and elements considered to classify are mainly focused to accomplish the problem:

2.1 Prioritization Goals

2.1.1 Fault Detection

Fault detection is the main aim here and hence we prioritize test suites in terms of the modules which tended to fail in the past. The goal here is to increase the rate of fault detection.

2.1.2 Code Coverage

Prioritizing of test cases is based upon increasing cost of coverage of code components or code features specified in requirement specification. The goal here is to increasing the code coverage.

2.1.3 Confidence in Reliability

Confidence of testers is considered as a parameter and test cases are prioritized such as to increase the trust of tester upon the system under test.

2.1.4 Correlated Fault Detection

Specific code changes can revoke faults which were solved in the past .The goal here is to prioritize test cases for early detection of such faults which occur due to correlated changes in code[5].

3. Classification Elements

3.1 Techniques Based on Customer Requirements

Requirement phase documents play a vital role to prioritize test cases in this method. Many weight factors such as customer assigned priorities; requirement volatility and requirement complexity govern these techniques.

3.2 Techniques Based on Coverage

The criterion which governs the prioritization in these techniques is coverage of various features such as requirement coverage, total requirement coverage, additional requirement coverage and statement coverage.

3.3 Techniques Based on Cost

The criterion which governs the prioritization in these techniques is cost effectiveness. Factors include cost of analysis and cost of prioritization.

3.4 Techniques Based on Test History

Test execution history forms the basic criteria of these techniques to prioritize test cases, such techniques use the previous performance of the test cases to decide whether the previous execution sequence should be repeated or not.

To accomplish the objectives of prioritization of test cases some techniques are suggested which provides adequate knowledge to prioritize test cases. Also, prioritization techniques are classified according to goals of prioritization and some basic factors governing prioritization criteria. Some other ways in which prioritization techniques can be classified are version based techniques; which are based on specific versions of software, Control techniques; which is realized by optimal and random ordering of test cases.

In the next section the survey is conducted for various popular test case prioritization techniques like RL-Based HMM method and the performance of the methods is evaluated based upon AFPD, effect size[5,6].

4. Different Test Case Prioritization Techniques used for Regression Testing

4.1 Test Case Prioritization using Extended Diagraphs

In [7]Recommended another fault-based test case prioritization strategy to advance fault-uncovering test cases in Model Based Testing (MBT) strategies. They enhanced the fault location rate a measure of how rapidly a test suite can identify faults amid testing in situations, for example, regression testing. They proposed an amplified digraph model as the premise of this new method. The model is acknowledged utilizing a novel Reinforcement Learning (RL) and Hidden Markov Model (HMM) based procedure which can organize test cases for regression testing. They introduced a strategy to instate and train a HMM based upon RL ideas connected to an application's digraph model. The model organizes test cases based upon forward probabilities, another test case prioritization technique. What's more, they additionally propose an option way to deal with organizing test cases as indicated by the measure of progress they cause in applications. To assess the viability of the proposed methods, they did probes Graphical User Interface (GUI) based applications and contrast the outcomes and best in class test case prioritization approaches. The exploratory results demonstrate that the proposed procedure can recognize faults ahead of schedule inside test runs.

4.2 JUPTA

In [8]Proposed JUnit test case Prioritization Techniques working in the Absence of scope information (JUPTA), which separates the static call charts of JUnit test cases and the program under test to assess the limit of each test case to fulfill code scope, and a short time later plans the demand of these test cases considering those evaluations. To survey the reasonability of JUPTA, they coordinated a correct review on 19 interpretations of four Java programs reaching out from 2K-80K lines of code, and differentiated a couple of varieties of JUPTA and three control frameworks, and a couple of other existing component scope based test case prioritization techniques, assessing the limits of the methodology to manufacture the rate of weakness acknowledgment of test suites. Their results show that the test suites worked by JUPTA are more convincing than those in discretionary and untreated test arranges similarly as lack acknowledgment sufficiency. Regardless of the way that the test suites created by component scope based methodology hold blemish area ampleness preferences, the lack ID suitability of the test suites delivered by JUPTA is close to that of the test suites made by those methodologies, and the inadequacy revelation feasibility of the test suites worked by some of JUPTA's varieties is better than that of the test suites made by a couple of those frameworks.

4.3 Modular Based Multiple Test Case Prioritizations

In [9]recommended another procedure to divide project into various modules then once after completion of testing by arranging test cases in accordance of modules, they are joined to form the original program.

4.4 A Clustering-Bayesian Network Based Approach for Test Case Prioritization

In [10]the approach is a crossover relapse test case prioritization procedure which means to accomplish better prioritization by joining code scope based clustering approach with Bayesian to discourage the effect of those comparable test cases having basic code scope. After investigating two java codes intentionally faulted the test comes about demonstrated that the proposed approach is promising.

4.5 System Level Test Case Prioritization

In [11]proposed organizes the framework test cases taking into account the elements: client need, dynamic requirements, usage unpredictability, and ease of use, application stream and Fault effect.

4.6 PORA

In [12]PORA guides explore prioritization by upgrading the separation between the sorted out test suite and a chain of command of scattering of test data information. Their examination shows that PORA analyze prioritization strategies are as suitable as, if not more feasible than, the aggregate covetous and extra eager methods, which use code scope data. Also, the examination exhibits that PORA systems are more stable in suitability than the others.

4.7 History-Based Test Case Prioritization

In [13]proposes an approach which focuses on how a particular test case performed previously and based upon that it is selected for the current testing by ranking its effect.

4.8 Oracle Centric Test Case Prioritization

In [14]proposed a methodology for arranging test cases that considers the effect of test oracles on the feasibility of testing. Their framework works by first catching the flood of data from variable assignments to test oracles for each analysis, and after that arranging to cover factors using the most restricted ways possible to a test oracle.

4.9 RDCC

In [15]introduces a prioritization framework, which takes SRS, plan structures, code and test cases as input and gives a recommended list of experiments utilizing their combined information as output. This framework is checked with various software and the results were found satisfactory.

4.10 Effective Input Based Randomized Test Case Prioritization

In [16]suggests a peculiar group of techniques which are based on random analysis mostly based on tree. These can be applied on programs with no historical record. Mathematical analysis of these techniques suggests that their performance is better than many greedy and coverage based techniques.

4.11 A similarity-based approach for test case prioritization

In [17]specifies a set of measurements that forecast the test case reliability comparing to the previously failing test cases. The results of their work represents that their proposed approach is better than many current test case measures.

4.12 Jtop

In [18]Jtop statically investigates the SUT and relevant JUnit test cases to carry the mentioned undertakings: regression test case selection, reducing test suite and TCP.

4.13 G-Rank Test

In [19]Suggests G-Rank Test, a strategy which investigates the use of result of output as a parameter for selecting test cases which triggers the most complex parts of the code and is simple to divide whenever a change happens.

4.14 Adaptive Random Test Case Prioritization

In [20]proposes a group of ART techniques which are dependent on code coverage. The advantage of this method is less cost at nearly the same output.

4.15 Time-Aware Test-Case Prioritization Using Integer Linear Programming

In [21]proposes a technique whose results suggests that their techniques are better than all others for the SUT and suggesting that it is best when there is no time constraints.

5. Performance Evaluation

In this section, different approaches are compared with the different parameters such as APFD, effect size and statistical testing.

Table 1 shows the comparison of APFD percentage for Universal Password Manager(UPM) GUI calculated by various methods. The methods taken into consideration are RL-Based HMM, Accumulated Q Value, Additional Statement Coverage, Random Control Operation, Worst Control Operation, and Optimal Control Operation[21]. The APFD of the above mentioned methods is 0.68, 0.61, 0.61, 0.5, 027, and 0.73 respectively. Optimal method is having highest APFD and suggested RL Based HMM is the next best which shows that its performance is best for detecting faults. APFD is calculated by the formula defined by as:

$$APFD(T') = 1 - \frac{TF_1 + TF_2 + TF_m}{nm} + \frac{1}{2n} \quad (1)$$

Here in equation 1 T' is a prioritized test suite, T is a test suite containing n test cases and F is a set of m faults revealed by T.

Table 1 shows the comparison of standard deviation from expected APFD of various methods. The methods taken into consideration are RL-Based HMM, Accumulated Q Value, Additional Statement Coverage, Random Control Operation, Worst Control Operation, and Optimal Control Operation. The standard deviation of the above mentioned methods is 0.00012, 0.00472,

Table 1. Comparison of methods based on APFD and deviation

Methods	APFD	Standard Deviation
RL-Based HMM	0.68	0.00012
Accumulated Q Value	0.62	0.00472
Additional Statement Coverage	0.61	0.0046
Random	0.5	0.0512
Worst	0.27	Nil
Optimal	0.73	Nil

Table 2. Effect size comparison with RL-based HMM

Comparison of Methods	T-statistic	Cohen's d
RL-Based HMM vs. Accumulated Q Value	3.16	1.63
RL Based HMM vs. Additional Statement Coverage	2.23	1.15
RL Based HMM vs. Random	5.3	2.77
RL Based HMM vs. Worst	6.4	3.34
RL Based HMM vs. Optimal	–2.24	–1.15

0.0046, and 0.05125,0,0 respectively. Again standard deviation of optimal method is zero which means its APFD value is expected next best is again RL Based HMM whose standard deviation is quite low which shows it is apt for detecting faults.

Table 2 shows the comparison between RL-Based HMM and other methods for Cohen's d and t-statistic which are parameters for effect size and statistical analysis. The table clearly shows RL-Based HMM is better in both the parameters when compared to other methods, only optimal method is better.

6. Conclusion

Some of the test case prioritization issues in regression testing are addressed and the techniques to overcome them are surveyed. While some of the techniques mentioned are traditional and based on basic prioritization goals where as some techniques use complex methods to optimize prioritization. These various ideas using the optimization techniques are brought in this paper and their advantages and limitations were discussed. This survey brings several ideas regarding various optimal prioritization techniques which mainly focus on coverage and faults, while some methods did not cover both the areas satisfactorily but they did it for some specific applications. In future there is scope for further improvement in prioritization techniques covering all applications. The other area of improvement can be to find best prioritization sequence and dealing with threats to validity of these methods.

7. References

1. Rothermel G, Harrold MJ. A safe, efficient regression test selection technique. Association for Computing Machinery (ACM) Transactions on Software Engineering and Methodology (TOSEM). 1997 Apr; 6(2):173–210. Crossref
2. Pravin P. Effective test case selection and prioritization in regression testing. Journal of Computer Science. 2013; 9(5):654–9. Crossref
3. Qu B, Nie C, Xu B. Test case prioritization based on test suite design information. Chinese Journal of Computers. 2009; 31(3):431–9. Crossref
4. Ansari A, Khan A, Khan A, Mukadam K. Optimized regression test using test case prioritization. Procedia Computer Science. 2016; 79:152–60. Crossref
5. Sumar G. An efficient method to achieve effective test case prioritization in regression testing using prioritization factors. Asian Journal of Information Technology. 2012; 11(5):169–80. Crossref
6. Chen X, Chen J, Ju X, Gu Q. Survey of test case prioritization techniques for regression testing. Journal of Software. 2014; 24(8):1695–712. Crossref
7. Muthusamy TKS. A new effective test case prioritization for regression testing based on prioritization algorithm. Indian Journal of Asian and Information Science. 2014 Jan; 6(7):21–6. Crossref
8. Kire K, Malhotra N. Study of test case selection and prioritization. International Journal of Computer Applications. 2014 Jan; 85(5):28–30. Crossref
9. Beena R, Sarala S. Code coverage based test case selection and prioritization. International Journal of Software Engineering and Applications (IJSEA). 2013 Nov; 4(6):39–49.
10. Catal C, Mishra D. Test case prioritization: a systematic mapping study. Software Quality Journal. 2012 Jul 26; 21(3):445–78. Crossref
11. Gokce N, Eminli M. Model-based test case prioritization using neural network classification. Computer Science and Engineering: An International Journal. 2014 Feb; 4(1):15–25.
12. Badhera U, Biswas D. Test case prioritization using fuzzy logic based on requirement prioritizing. International Journal of Computer Science and Applications (IJCSA). 2013; 3(2):23–9.
13. Panigrahi C, Mall R. Model-based regression test case prioritization. SIGSOFT Software Engineering Notes. 2010 Nov; 35(6):1–7. Crossref
14. Singhal H, Tyagi K. An evolution of test case prioritization techniques. International Journal of Computer Applications. 2015 Nov; 130(1):33–7. Crossref
15. Chandu P, Sasikala T. Implementation of regression testing of test case prioritization. Indian Journal of Science and Technology. 2015 Apr; 8(S8):290–3. Crossref
16. Wu X, Jiang R, Michael Q, Zhang Z, Li S. Test case prioritization using hyperlink ranking - a graph theory based approach. International Journal of Research in Engineering and Technology. 2013 Nov; 2(11):29–32. Crossref

17. Singh Y, Kaur A, Suri B. Test case prioritization using ant colony optimization. SIGSOFT Software Engineering Notes. 2010 Jul; 35(4):1–7. Crossref
18. Rajal JS, Sharma S. A review on various techniques for regression testing and test case prioritization. International Journal of Computer Applications. 2015 Apr; 116(16):8–13.
19. Pathania Y, Kaur G. Role of test case prioritization based on regression testing using clustering. International Journal of Computer Applications. 2015 Apr; 116(19):7–10. Crossref
20. Chen G, Wang P. Test case prioritization in a specification-based testing environment. Journal of Software. 2014; 9(8):1–9.
21. Li Z, Harman M, Hierons R. Search algorithms for regression test case prioritization. Institute of Electrical and Electronics Engineers (IEEE) Transactions on Software Engineering. 2007 Apr; 33(4):225–37. Crossref