Test Case Prioritization: Various Techniques–A Review

Sunny Kumar, Sheena Singh

Abstract— Regression testing is extremely very expensive process for software development and maintenance. Whenever software is modified then re-executing all the test cases that have previously tested to ensure that there no new errors have been introduced in the code due to modification. For this researchers can prioritize the test cases to increase the effectiveness of the testing process and to reduce the cost of regression testing. With the help of test case prioritization researchers can schedule test cases in a test suite by ordering them in such a way that those with higher priority according to some criterion are executed earlier than those with lower priority. In this paper we present the review of some popular test case prioritization techniques like Average Percentage of Fault Detection (APFD), Severity of Faults, Mutation Faults, Ordered Sequence of Program Elements, and Search Algorithm are some popular techniques of test case prioritization with their pros and cons.

Index Terms— APFD, Fault Localization, Fault Severity, Mutation Faults, Model Checker, Search Algorithm, Test Case Prioritization

1 Introduction

IN the software development life cycle regression testing is very expensive process. But it ensures that the software program will work according to its specification after changes have been made to it [6]. About 50% of the total software cost is consumed in the testing phase [4]. With the help of regression testing the test cases are prioritize by the engineers and they run those test cases earlier which have more important than other test cases by some criteria. The main objective of the test case prioritization is fault detection in the software program. Nowadays, in software testing; there are so many techniques proposed by the researchers for test case prioritization to prioritize the test cases. In this paper we are discussing various Test Case Prioritization (TCP) techniques which are mostly used by the researchers these days, such techniques are: Ordered Sequence of Program Elements [10], Mutation Faults [9], and Fault Localization [7], Fault Severity [3], Model Checker [17] etc.

2 Review Of Literature

2.1 Fault Severity

On the basis of requirement the prioritization is done for requirement on the base of severity of faults. The Total Severity of Faults Detected (TSFD) is the summation of severity measures of all faults identified for a product is given in the below equation (1):

$$\text{TSFD: } \sum_{i=1}^{i=n} \text{SM (severity measure)} \qquad (1)$$

Where:
n represents the total number of faults identified for the product [1].

To improve the rate of fault detection at system level for severe faults a value driven test case prioritization for requirement is used. There are three prioritization factors are used for the requirements customer priority, implementation complexity and volatility [4].

2.2 Fault Localization

Fault localization technique is used to locate faults by using the execution information collected during testing phase. It is also one of the major tasks in debugging [7]. Researchers can define the metric Fault Localization Successful Percentage (FLSP) as; it is the ratio of the number of effective test suites in a test suite pool over the size of the test suite pool with respect to a fault localization technique and a given faulty program [5].

2.3 Mutation Faults

Mutation testing comes from the idea of mutation faults. It is a technique which evaluates the capability of a test suite of a program by placing simple syntax code changes into the software program and checking whether the test suite can detect these changes in the code [9]. The rate of fault detection for JUnit test suites in test case prioritization is assessed relative to mutation faults, but the results vary with the test suites and with the numbers of faults detected [12].

2.4 Ordered Sequence of Program Elements

Execution frequencies are main concept behind the idea of ordered sequence of program elements. With the help of this technique can detect bugs in loop more quickly. An algorithm farthest - first is adopted to maximize diversity of test cases. This algorithm selects the next test case which is farthest away from the already selected test suite. And the process goes on till all the test cases are re-ordered [10].

- *Student, Sunny Kumar is currently pursuing master's degree program in Computer Science and Engineering inLovely Professional University, Phagwara, India, PH-07307123485. E-mail: sunnysethi90@gmail.com*
- *Assistant Professor, Sheena Singh is currently working in department of Computer Science and Engineering in Lovely Professional University, Phagwara, India, PH-09646689681. E-mail: sheena.15740@lpu.co.in*

2.5 Average Percentage of Fault Detection (APFD)

An algorithm is used by the researchers for test case prioritization, which calculates the average faults found per minute. They calculate APFD only when we have the primarily knowledge about the available faults. The calculation of APFD is by taking the weighted average of test suite. Formula to calculate APFD is in the below equation (2):

$$\text{APFD}: 1 + \frac{(TF_1 + TF_2 + \ldots\ldots + TF_{m)}}{(nm)} + \frac{1}{2n} \qquad (2)$$

Where:

T:- the test suite containing n test cases.

m:- m is the no. of faults.

TF:-The position of the first test in T that exposes faults i [3].

APFD measures the rate of fault detection per percentage of test suite execution [6]. By measuring the APFD value researchers can see how quickly the faults are identified for a given test suite set [2]. The test cases are prioritized on the base of rate of fault detection and fault impact [4].

2.6 Model Checker

There are four prioritization methods for model checker such as Coverage Prioritization, FEP Prioritization, Property Prioritization, and Optimal Prioritization. It is very necessary to recreate test-cases as models, which allows analysis with concern to certain properties. It can be easily done by basing the transition relation of all variables on a special state-counting variable. To determine prioritization with model checker researchers base the techniques on a functional model of the program to test [17].

2.7 Search Algorithm

For regression testing Hill climbing, Greedy algorithm, Genetic algorithms, Additional greedy algorithm, and optimal algorithm are used for the sequencing problem in test case prioritization. The various search algorithms are:

2.7.1 Greedy Algorithm

Greedy Algorithm helps to reach some goal and its search seeks to minimize the estimated cost. It is very simple and inexpensive in execution time and in implementation situations where the results are of high quality [16].

2.7.2 Additional Greedy Algorithm

Additional Greedy Algorithm helps in iteratively selection on the maximum weight element of the problem from that part that has not already been consumed by previously selected elements [16].

2.7.3 Optimal Algorithm

The 2-Optimal Algorithm helps in the updates coverage of information for each unselected test case following the choice of each pair of test cases. This approach has been found to be effective and fast [16].

2.7.4 Hill Climbing algorithm

Hill Climbing algorithm is very easy and helps in searching and produces suboptimal results that are only locally optimal. It is very simple and cheap to implement and execute [16].

2.7.5 Genetic Algorithms

Genetic Algorithms helps to represent a class of adaptive search techniques based on the processes of natural genetic selection according to Darwinian Theory of biological evolution [16]. In Genetic Algorithm the initial population is automatically produced and the evaluation of the set of candidate solution has been done. The total code coverage is the stop criteria for this approach [11]

2.8 Other Techniques

There are many other techniques for test case prioritization such as Empirical study [13],[14],[15], Coverage based [14], Modified Condition/Decision Coverage [8], and White Box Testing, Black box testing, etc for test case prioritization but not discussed in this paper. Table I represents the summary of this paper.

Table I: Summary of Test Case Prioritization Techniques with Their Pros and Cons

S. No.	Technique	Key Idea	Advantage	Disadvantage
1.	Fault Severity	Base on requirement specification	1. It can improve the software quality. 2. The faults are produced quickly with high severity [1]. 3. It can improve the rate of fault detection [4]. 4. Requirements volatility is a significant factor in industrial projects.	1. It does not remove the induced factor of requirements volatility. 2. Project scope is limited [4].
2.	Fault Localization	Based on execution information of fault loca-	1. A postmortem analysis approach [5].	1. Insufficient in the effective support of statis-

		tion	2. Faster failure exposes [7].	tical fault localization [5]. 2. The subsequent fault localization may suffer [7].
3.	Mutation Faults	Based on changes in program code	1. Rate of fault detection is improved [9].	1. Cost reduction is still not significant.
4.	Ordered Sequence of Program Elements	Based on execution frequencies of the program element	1. Bugs are detected quickly in loops. 2. Cost effective approach [10].	1. Still it's not much effective approach.
5.	APFD	Based on average faults found per minute	1. Rate of faults detection is easy at system level [3].	1. This technique not much more efficient in fault detection.
6.	Model Checker	Based on functional model of program test	1. Prioritization is efficiently applied on the time of creation of test cases [17].	Many factors are still not included such as: 1. Actual test case execution costs. 2. The costs of potential faults [17].
7.	Search Algorithm	Based on code coverage	1. Efficient and Flexible. 2. Size of the program does not have any effect on test case prioritization [11].	1. Still cannot solve large number of test case [11]. 2. It possibly produces different result [16].

3 CONCLUSION

From the above we can conclude that there are so many techniques that are used for test case prioritization. Each one of technique has their own pros and cons. But according to our requirement a tester can use any technique.

ACKNOWLEDGMENT

I would like to place on record my deep sense of gratitude of Ms. Sheena Singh, Assistant Professor at Lovely Professional University, for their valuable time and useful suggestions which are responsible for the work produced here.

REFERENCES

[1] Kumar, Dr Varun. "Sujata and M. Kumar,"Test Case Prioritization Using Fault Severity"." *IJCST* 1, no. 1 (2010): 67-71.

[2] Kavitha, R., and N. Sureshkumar. "Test Case Prioritization for Regression Testing based on Severity of Fault." *International Journal on Computer Science and Engineering* 2, no. 5 (2010): 1462-1466.

[3] Srivastava, Praveen Ranjan. "Test case prioritization." *Journal of Theoretical and Applied Information Technology* 4, no. 3 (2008): 178-181.

[4] Srikanth, Hema, Laurie Williams, and Jason Osborne. "System test case prioritization of new and regression test cases." In *Empirical Software Engineering, 2005. 2005 International Symposium on*, pp. 10-pp. IEEE, 2005.

[5] Jiang, Bo, and W. K. Chan. "On the Integration of Test Adequacy, Test Case Prioritization, and Statistical Fault Localization." In *Quality Software (QSIC), 2010 10th International Conference on*, pp. 377-384. IEEE, 2010.

[6] Raju, S., and G. V. Uma. "Factors Oriented Test Case Prioritization Technique in Regression Testing using Genetic Algorithm." *European Journal of Scientific Research* 74, no. 3 (2012): 389-402.

[7] Jiang, Bo, Zhenyu Zhang, T. H. Tse, and Tsong Yueh Chen. "How well do test case prioritization techniques support statistical fault localization" In *Computer Software and Applications Conference, 2009. COMPSAC'09. 33rd Annual IEEE International*, vol. 1, pp. 99-106. IEEE, 2009.

[8] Jones, James A., and Mary Jean Harrold. "Test-suite reduction and prioritization for modified condition/decision coverage." *Software Engineering, IEEE Transactions on* 29, no. 3 (2003): 195-209.

[9] Do, Hyunsook, and Gregg Rothermel. "On the use of mutation faults in empirical assessments of test case prioritization techniques." *Software Engineering, IEEE Transactions on* 32, no. 9 (2006): 733-752.

[10] Wu, Kun, Chunrong Fang, Zhenyu Chen, and Zhihong Zhao. "Test case prioritization incorporating ordered sequence of program elements." In*Automation of Software Test (AST), 2012 7th International Workshop on*, pp. 124-130. IEEE, 2012.

[11] Kaur, Arvinder, and Shubhra Goyal. "A genetic algorithm for regression test case prioritization using code coverage." *International Journal on Computer Science and Engineering* 3, no. 5 (2011): 1839-1847.

[12] Do, Hyunsook, and Gregg Rothermel. "A controlled experiment assessing test case prioritization techniques via mutation faults." In *Software Maintenance, 2005. ICSM'05. Proceedings of the 21st IEEE International Conference on*, pp. 411-420. IEEE, 2005.

[13] Rothermel, Gregg, Roland H. Untch, Chengyun Chu, and Mary Jean Harrold. "Test case prioritization: An empirical study." In *Software*

Maintenance, 1999.(ICSM'99) Proceedings. IEEE International Conference on, pp. 179-188. IEEE, 1999.

[14] Rothermel, Gregg, Roland H. Untch, Chengyun Chu, and Mary Jean Harrold. "Prioritizing test cases for regression testing." *Software Engineering, IEEE Transactions on* 27, no. 10 (2001): 929-948.

[15] Elbaum, Sebastian, Alexey G. Malishevsky, and Gregg Rothermel. "Test case prioritization: A family of empirical studies." *Software Engineering, IEEE Transactions on* 28, no. 2 (2002): 159-182.

[16] Li, Zheng, Mark Harman, and Robert M. Hierons. "Search algorithms for regression test case prioritization." *Software Engineering, IEEE Transactions on* 33, no. 4 (2007): 225-237.

[17] Fraser, Gordon, and Franz Wotawa. "Test-case prioritization with model-checkers." In *25th conference on IASTED International*. 2007.

Sunny Kumar received his B.Tech degree in Information Technology from Lovely Professional University, Phagwara, Punjab, India in 2011. Now he is doing M.Tech in Computer Science and Engineering from Lovely Professional University, Phagwara, Punjab, India. His research interests include Software Engineering.

Sheena Singh received her B.Tech degree in Computer Science and Engineering from Lovely Institute of Technology from Phagwara, Punjab, India in 2008, and M.Tech in Computer Science and Engineering from Rayat Institute of Engineering and Technology from Ropar, Punjab, India in 2011. Now she is working as an Assistant Professor with Department of Computer Science and Engineering at Lovely Professional University, Phagwara, Punjab, India. Her research interests include Software Engineering.