ISSN 2278-3091
Volume 8, No.4, July – August 2019
International Journal of Advanced Trends in Computer Science and Engineering
Available Online at http://www.warse.org/IJATCSE/static/pdf/file/ijatcse74842019.pdf
https://doi.org/10.30534/ijatcse/2019/74842019

Comparative Analysis of Regression Test Case Prioritization Techniques

Omdev Dahiya[1], Kamna Solanki[2], Sandeep dalal[3]
[1,2]University Institute of Engineering and Technology, Maharshi Dayanand University, Rohtak-124001, India
[3]Department of computer science and applications, Maharshi Dayanand University, Rohtak-124001, India
[1]Omdahiya21792@gmail.com, [2]Kamna.mdurohtak@gmail.com

ABSTRACT

Software testing is an activity performed for evaluating the system by various testing techniques. Regression test case selection selects those test cases which focuses on the modified part of the software. It generally filters test cases instead of removing them. Test case minimization eliminates certain test cases thus, reducing the number of test cases and lays stress on using a subset of the test suite from an economy point of view. This sometimes reduces the rate of fault detection. Elimination of test cases is not performed in test case prioritization, rather it arranges them according to priority and they are thus executed with higher priority cases first followed by the lower priority ones. Test case prioritization can be search-based or fault-based or coverage based. They may also be risk-based, fault-based, history-based or requirement based.

This paper has presented a comparative analysis of various TCP approaches according to the prioritization objectives framed for this study. The aim is not to show that a particular approach has some limitations as the performance of an approach varies on multiple factors such as testing scenarios, testing needs, size of the program upon which they were applied and the testing environment. The factors governing the need for the development of a certain approach are fast execution of test cases or early fault detection or making testing cost-effective, although researchers try to achieve most of them if not all. The approaches developed by numerous researchers are comparatively studied and results are presented systematically.

Key words: Regression testing, Software failure, Software quality, Software testing, Test case prioritization.

1. INTRODUCTION

In the present scenario, the role of software can be traced from every device associated with human beings to life-saving devices in the medical field, in space technology and aviation-related devices also. It can be said that software is omnipresent. Therefore, so much of stress is laid out on the quality and reliability of the software [1]-[3]. Testing plays a vital role in examining whether the developed software product meets the desired standards or not [4]-[5]. Many techniques for testing the software were developed by the researchers. Exhaustive testing is used to execute every test case designed for testing. But its drawback is that it is not practically feasible owing to time, resource and cost constraint [6]-[8]. A recent software glitch in Boeing 737 max caused a huge loss of human life and money in which a software problem has resulted in a faulty sensor reading which forced the plane nose down leading to two fatal crashes killing altogether 346 people on board [9]. Another example is of Microsoft windows operating system in which software vulnerability popularly known as "EternalBlue" has resulted in a huge amount of loss and a serious lesson to remain cautious of releasing security patches and to spread awareness about cybersecurity [10].

The Organization of this paper is as follows: In Section 2 regression testing and techniques for test case prioritization are discussed in detail. Section 3 presents the need to conduct this study and discussed the prioritization objectives framed for this paper. Section 4 presents an in-depth analysis of different techniques for test case prioritization, which is subsequently followed by a table showing comparative analysis. Section 5 discusses conclusion and future work. The last section discusses the references included in this study.

2. REGRESSION TESTING

One of the popular approaches in testing is regression testing which is performed to check whether no new faults have crept in the unmodified code after some part of the source code is modified [11]-[12]. An approach in regression testing is retest-all the test cases on the modified program, but it is not practically feasible [13]. So, developers have to maintain the balance between the quality of the software product and the cost accomplished in testing as customer satisfaction is the ultimate goal of a software development organization [1], [14]-[15].

Therefore, to make the regression testing effective, various strategies are developed which are Test Case Selection (TCS), Test Suite Minimization (TSM), Test Case Prioritization (TCP) [16]. TCS selects those test cases which may focus on testing the modified part of the software. TSM discards redundant test cases to reduce the test suite size. Both

TCS and TSM approaches may lead to a decrease in fault detection rate as claimed by different researchers [17]-[18]. This is the driving force which motivates the researchers to work on TCP approaches which neither selects nor discards test cases but executes them in a certain priority way to increase detection of faults. Researchers have proposed numerous test case prioritization approaches (figure 1). [16], [19]-[24].

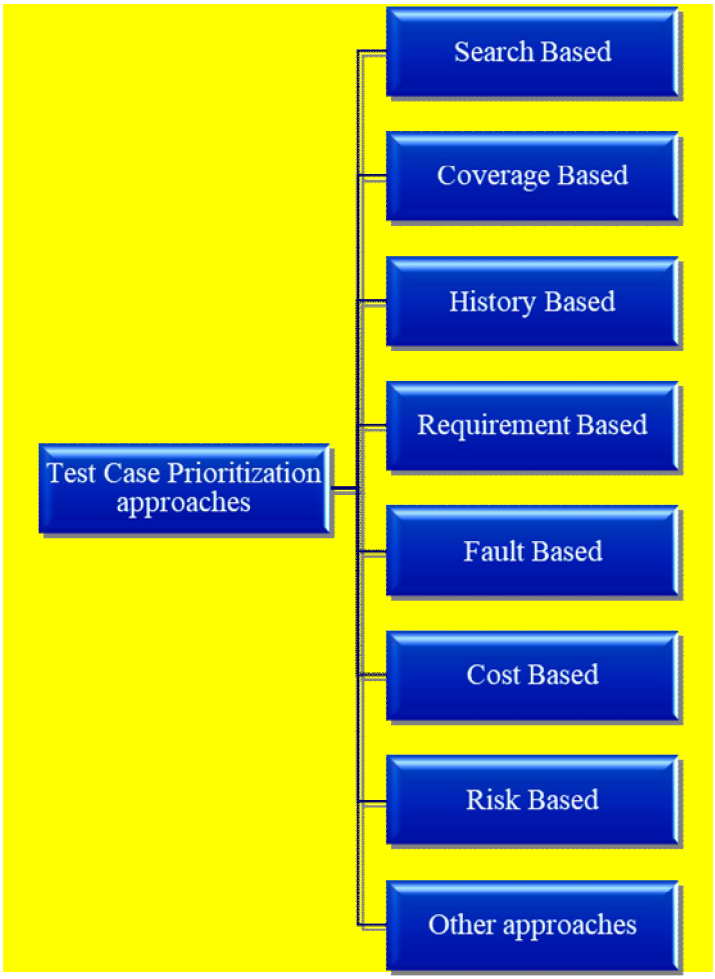

Figure 1: Categorization of TCP approaches. [19], [24]

The goal of TCP can be manifold such as enhancing the rate of fault detection or reducing the time and cost incurred in the mechanism of prioritization or to increase the capture of high priority requirements.

3. THE MOTIVE OF THIS STUDY

This paper has worked to comparatively analyze various TCP approaches for regression testing. This study is not performed to discuss advantages or limitations of various approaches, rather the motive of this study is to provide a detailed analysis of the existing approaches by comparing them with the prioritization objectives framed for this study.

For this paper, the articles were searched from Scopus and Google Scholar database and due to this all the major venues were covered such as Elsevier, Springer, IEEE, ACM, etc. For searching the papers, a search string is devised after various experiments, adding and removing keywords in it and using "AND", "OR" operations. The search string is < "Software testing" and "Regression testing" and "Test case prioritization">. In the next step, collectively from both the databases, a large number of studies were retrieved and after following the guidelines by Kitchenham [25] and the defined procedure along with inclusion and exclusion criteria, finally we were able to finalize 85 studies for inclusion in this study which have fulfilled our defined prioritization objectives well. Accordingly, a table is then made which shows a comparative analysis of the different approaches (figure 2).

PRIORITIZATION OBJECTIVES	EXPLANATION
PO1	Whether fault detection capability is enhanced by the proposed technique and/or whether historical information of test cases is used to generalize the result.
PO2	Whether the overall cost of regression testing has been reduced and does the approach resulted in a cost-effective one.
PO3	Whether execution time was reduced along with an increased rate of fault detection and/or whether maximum user requirements were captured.
PO4	Whether maximum code coverage has been achieved in terms of branch/statement/function so that fault can be detected early in modified code.
PO5	Whether risk factor associated with particular test cases are taken into account for prioritizing test cases and/or whether the proposed technique has also shown effectiveness in testing real-world systems/Industrial study/web applications like GUI/web-based applications.

Figure 2: Prioritization objectives for this study.

The aim of designing the prioritization objectives is to comparatively analyze different techniques designed by the researchers based on these objectives. Every technique executes uniquely according to the provided testing environment, the size of programs upon which they were

incorporated and based on testing needs.

4. IN-DEPTH ANALYSIS OF TEST CASE PRIORITIZATION APPROACHES

This section will provide a detailed review of the existing techniques which covers the basic aspects and factors based on which these approaches were proposed by the researchers. This will facilitate the readers to have a better understanding of the different techniques for test case prioritization.

4.1 Search-based test case prioritization

These approaches have followed nature inspired metaheuristic algorithms to solve complex problems in computer science. These include Genetic algorithm [26]-[34], Ant colony optimization [35]-[39], and others [40]-[42]. Opinion for the usage of these approaches varies according to different researchers, such as shown by Li *et al.*, various factors play a role here such as fitness function, test suite, input criteria, etc [43]. Still, these approaches were favored by various researchers due to its advantages.

4.2 Coverage-based test case prioritization

These approaches aim to provide maximum coverage of the branch/statement/function to enhance the rate of fault detection. In this, there is direct inspection of source code [44]. After the evaluation of the criteria, these approaches are further categorized as additional and total coverage based [16]. These techniques were followed by numerous researchers [23], [45]-[52]. An enormous study is conducted by various researchers and logical reasoning is provided by them that greater effectiveness is achieved by greater coverage. Although, it is one of the criteria of test case prioritization which may or may not leads to cover all the faults.

4.3 History-based test case prioritization

These techniques use a history of test case execution for prioritizing the test cases. Kim and porter proposed a history-based approach for TCP [53]. This approach is further extended in the research conducted by Khalilian *et al.*, in which it is concluded that the fault detection rate is increased if test cases are prioritized by using historical test data [54]. The result of this work is shown using APFDc metric. This approach is favored by several researchers [55]-[59]. Though this being an effective approach, it has few shortcomings such as availability of historic information is limited and information about a defect is also available is a small number.

4.4 Requirement-based test case prioritization

These techniques state that as requirements are the building blocks of a system, so important test cases can be classified utilizing requirement information. A technique called “PORT- prioritization of requirements for test" is developed by Srikant *et al.*, in which a criterion is given to provide the value of software requirement importance [60]. In a conducted study by Muthusamy, various perspectives of requirement were given and they have proved their result to be effective in terms of rate of fault detection [61]. In the research performed by Srikant *et al.*, it is concluded that if two or more factors are taken in a combination, then testing is said to be much more effective [62]. Owing to the trends, many researchers have worked on these approaches [63]-[74], [92]-[93], [109]. It shows how this area is gaining popularity to deliver a quality product to the customer within the stipulated budget and time.

4.5 Fault-based test case prioritization

The approach of these techniques is to specifically target certain faults by arranging test cases in a certain sequence. As stated by Rothermel *et al.*, during the execution of a particular statement specific faults can be discovered and possibilities of discovering other errors are also high while executing a statement [75]. Various approaches for enhancing the detection of faults is given by many researchers so that testing can be termed as an effective one [76-79].

4.6 Cost-based test case prioritization

The reason behind working on the cost model is to make the testing process cost-effective one. A cost-based model is proposed by Huang *et al.* in the study conducted by them [80]. Malishevsky *et al.* presented a cost-based metric i.e APFDc [81]. Work was also performed by other researchers on a cost-based model so that the cost of testing can be reduced [82-85].

4.7 Risk-based test case prioritization

Work on risk-based approaches is started to target risk-prone elements in the software and if the risk is timely evaluated it will help in preventing further damage [86]. According to Stallbaum *et al.*, test cases are prioritized based on the risk associated with them and these test cases are generated from the activity diagram via a technique proposed by them [87]. In the study performed by Yoon, risky elements were assigned a risk exposure value which is originated from requirements and test cases are prioritized accordingly [88]. Many researchers have worked on these approaches as, if risk causes damage then a huge amount of loss is incurred by the software industry [89]-[91].

4.8 Other approaches for test case prioritization

Apart from the approaches discussed above, researchers have worked upon many other techniques such as model-based [94]-[100], web application-based [101]-[105] and even techniques to be applicable on real-world systems [44], [106]-[108]. For approaches to testing automation, model-based testing is gaining popularity among researchers as it is considered as a profitable option. In this, system information about architecture and data structure is required instead of source code and when compared to the actual system its execution is also fast. TCP techniques for testing web applications is also gaining popularity due to the increase in usage of the Internet. For real-world systems regression faults are only a few and difficult to locate, so researchers have also explored this sphere.

Table 1: Comparative analysis of the different techniques for test case prioritization

Author, Publication year and Reference	TCP technique/ Algorithm/ Metric used to evaluate the proposed technique.	PO 1	PO 2	PO 3	PO 4	PO 5
Y. Lou *et al.*, 2015 [26]	Search-based, GA, APFD	✓	×	✓	✓	×
Maheswari and Mala, 2015 [27]	Search-based, GA	✓	×	✓	×	×
C. Catal, 2012 [28]	Search-based, GA	-	-	-	×	×
Yuan *et al.*, 2015 [29]	Search-based, GA, APSC	✓	×	×	✓	×
Deb *et al.*, 2002, [30]	Search-based, GA, ♈, △	-	-	-	✓	-
Kaur and Goyal, 2011 [31]	Search-based, GA, APFD	✓	×	✓	✓	×
Jun *et al.*, 2011 [32]	Search-based, GA, APBC	✓	✓	✓	×	×
Sabharwal *et. al.*, 2010 [33]	Search-based, GA	×	✓	×	-	-
Huang *et al.*, 2010 [34]	Search-based, GA, APFDc	✓	✓	×	×	×
Singh *et al.*, 2010 [35]	Search-based, ACO, APFD	✓	✓	✓	×	×
Gao *et al.*, 2015 [36]	Search-based, ACO, APFD	✓	×	✓	×	×
Suri and Singhal, 2011 [37]	Search-based, ACO, TC	✓	✓	✓	✓	×
Noguchi *et al.*, 2015 [38]	Search-based, ACO	✓	×	✓	×	×
Solanki *et al.*, 2016 [39]	Search-based, ACO, APFD, PTR	✓	×	✓	✓	×
Eghbali and Tahvildari, 2016 [40]	Others, APFD	✓	×	✓	✓	-
Mala *et al.*, 2009 [41]	Others, ABC	✓	✓	✓	×	×
Jeffrey and Gupta, 2006 [42]	Others, APFD	✓	×	×	✓	×
Li *et al.*, 2016 [43]	Search-based, APRCI	✓	×	✓	×	×
Nardo *et al.*, 2015 [44]	Search-based, Real-world systems, APFD	✓	✓	✓	✓	✓
Do and Rothermel, 2015 [45]	Coverage-based, APFD	✓	✓	×	✓	×
Do *et al.*, 2006 [46]	Coverage-based, APFD	✓	✓	✓	✓	×
Rothermel *et al.*, 2001 [47]	Coverage-based, APFD	✓	✓	×	✓	×
Kapfhammer and sffa, 2007 [48]	Coverage-based, CE	✓	✓	×	✓	×
Hao *et al.*, 2015 [49]	Coverage-based, APFD, APxC	✓	×	✓	✓	×
Fang *et al.*, 2012 [50]	Coverage-based, CI	✓	×	×	✓	-
Elbaum *et al.*, 2002 [51]	Coverage-based, APFD	✓	✓	×	✓	×
Nardo *et al.*, 2013 [52]	Coverage-based, APFD	✓	×	×	✓	✓
Kim *et al.*, 2002 [53]	History-based	✓	✓	×	×	×
Khalilian, 2012 [54]	History-based, APFD	✓	✓	✓	×	-
Park *et al.*, 2008 [55]	History-based, APFDc	✓	✓	✓	×	×
Kim and Baik, 2010 [56]	History-based, FATCP, APT	✓	✓	✓	✓	×
Fazlalizadeh *et al.*, 2009 [57]	History-based	✓	×	×	×	-

Aman *et al.*, 2012 [58]	History-based	✓	×	-	×	×
Megala, 2017 [59]	History-based, GA, APFD	✓	✓	✓	×	×
Srikanth *et al.*, 2005 [60]	Requirement-based, PORT, WPFD	✓	×	✓	✓	×
Muthusamy, 2014, [61]	Requirement-based, APFD	✓	✓	✓	×	×
Srikanth *et al.*, 2016 [62]	Requirement-based, ReBaTe, TSFD, APFD	✓	×	✓	✓	✓
Srikanth and Williams, 2005 [63]	Requirement-based, PORT, ASFD	✓	×	✓	×	×
Srikanth *et al.*, 2013 [64]	Requirement-based, PORT, APFD	✓	×	✓	✓	✓
Ma *et al.*, 2016 [65]	Requirement-based, APFD	✓	×	✓	×	✓
Arafeen and Do, 2013 [66]	Requirement-based, APFD	✓	×	✓	×	×
Krishnamoorthi and Mary, 2009 [67]	Requirement-based, RFV	✓	✓	✓	×	✓
Zhang *et al.*, 2007 [68]	Requirement-based, MRP_TC	✓	✓	✓	×	×
Ramasamy and Mary, 2008 [69]	Requirement-based, TPFD	✓	✓	✓	×	✓
Salem and Hassan, 2011 [70]	Requirement-based	✓	×	✓	✓	×
Kavitha *et al.*, 2010 [71]	Requirement-based	✓	×	✓	×	✓
Berander and Andrews, 2005 [72]	Requirement-based	-	×	✓	-	-
Salehie *et al.*, 2011 [73]	Requirement-based, APFD	✓	×	✓	×	✓
Dongoor, 2019 [74]	Requirement-based	-	-	✓	-	✓
Rothermel *et al.*, 1999 [75]	Fault-based, APFD	✓	×	×	✓	×
Yu and Lau, 2012 [76]	Fault-based, APFD, FATE	✓	×	✓	×	×
Solanki *et al.*, 2016 [77]	Fault-based, m-ACO, APFD, PTR	✓	×	✓	×	✓
Jiang *et al.*, 2012 [78]	Fault-based	✓	✓	✓	×	×
Solanki *et al.*, 2013 [79]	Fault-based, m-ACO, APFD	✓	×	✓	×	×
Huang *et al.*, 2012 [80]	Cost-based, GA, APFDc	✓	✓	×	-	-
Malishevsky *et al.*, 2006 [81]	Cost-based, APFDc	✓	✓	×	×	×
Malishevsky *et al.*, 2002 [82]	Cost-based	✓	✓	×	✓	×
Elbaum *et al.*, 2004 [83]	Cost-based, APFD	✓	✓	×	×	×
Smith and Kapfhammer, 2009 [84]	Cost-based, RFFS	✓	×	✓	✓	✓
Srikanth *et al.*, 2009 [85]	Cost-based, NAPFD	✓	✓	×	×	-
Srivastava, 2008 [86]	Risk-based	✓	✓	✓	×	✓
Stallbaum *et al.*, 2008 [87]	Risk-based, RiteDAP, APDP	✓	×	×	×	✓
Yoon, 2012 [88]	Risk-based, APFD	✓	✓	✓	×	✓
Hettiarachchi, 2014 [89]	Risk-based	✓	✓	✓	-	✓
Yoon and Choi, 2011 [90]	Risk-based	✓	✓	×	×	✓
Hettiarachchi *et al.*, 2016 [91]	Risk-based, APFD, PTRSW	✓	×	✓	×	✓
Uusitalo *et al.*, 2008 [92]	Requirement-based	-	-	✓	-	-
Mogyorodi, 2001 [93]	Requirement-based	×	×	✓	✓	×
Thomas *et al.*, 2014 [94]	Model-based, APFD	✓	✓	×	×	×
Korel *et al.*, 2005 [95]	Model-based, MLP	✓	×	✓	×	×
Korel *et al.*, 2007 [96]	Model-based, MLP	✓	×	✓	×	-
Korel *et al.*, 2008 [97]	Model-based, MLP	✓	✓	✓	×	×
Hemmati *et al.*, 2013 [98]	Model-based	✓	✓	✓	-	✓
Panigrahi and Mall, 2014 [99]	Model-based, APFD	✓	✓	×	×	✓
Panigrahi and Mall, 2010 [100]	Model-based	✓	✓	×	×	×
Sampath *et al.*, 2008 [101]	Web application-based, APFD	✓	✓	✓	×	✓
Memon and Xie, 2005 [102]	Web application-based	✓	×	✓	×	✓

Sprenkle, 2005 [103]	Web application-based	✓	✓	✓	-	✓
Bryce and Memon, 2007 [104]	Web application-based	✓	×	×	×	✓
Bryce *et al.*, 2011 [105]	Web application-based, APFD	✓	×	✓	×	✓
Haymar and Hla, [2008 [106]	Real-world systems, PSO	✓	×	×	✓	✓
Srivastava and Thiagarajan, 2002 [107]	Real-world systems, Echelon	✓	×	✓	×	✓
Lu *et al.*, 2016 [108]	Real-world systems	✓	-	-	-	✓
Vescan *et al.*, 2017 [109]	Requirement-based	✓	✓	✓	✓	×
Schwartz and Do, 2016 [110]	Cost-Based	✓	✓	×	×	×
Sujata and Purohit, 2015 [111]	Factors for selecting TCP technique	✓	✓	✓	✓	-

Following is a list of abbreviations used in this table-

GA- Genetic algorithm, ACO- Ant colony optimization, mACO- Modified ant colony optimization, APFD- Average percentage of faults detected, APSC- Average percentage of statement coverage, APBC- Average percentage of blocks covered, ϒ- Convergence metric, △- Metric to measure spread achieved in obtained solutions, TC- Time constraint, ABC- Artificial bee colony optimization, PTR- Percentage of test suite required for complete fault coverage, APRCI- Average percentage of requirement coverage improved, CE- Coverage effectiveness, FATCP- Fault aware test case prioritization, APT- Average percentage of fault detected parsing tool, APxC- Average percentage of x coverage; x-branch/decision/statement, CI- Convergence index, WPFD- Weighted percentage of fault detected, ReBaTe- Requirement based testing, TSFD- Total severity of fault detected, RBT- Requirement behavior tree, ASFD- Average severity of fault detected, RFV- Requirement behavior tree, TPFD- Total percentage of fault detected, MRP_TC- Rate of units of testing requirement priority satisfied per unit test case cost, FATE- Fault adequate test set sizE, RFFS- Reduction factor for size, NAPFD- Normalized average percentage of fault detected, PTRSW- Percentage of Total Risk Severity Weight, MLP- Most likely relative position, APDP- Average Percentage of Damage Prevented, RiteDAP- Risk based test case derivation and prioritization.

From the above table, it is quite clear that most of the studies have not fulfilled all the PO for this study. This does not show that they lag in their effectiveness. It is evident that if a particular approach has focused on maximum fault detection then it may or may not have shown promising results in terms of least execution time or with minimum cost. In the same way, if a particular approach is proposed by the researchers then they must not have tested it using industrial data set as well or is developed by capturing maximum user requirements. So, this study is of the view that every proposed technique has fulfilled its goals which are framed by the respective researchers for it. The aim of this study is fulfilled in terms of selecting and then comparatively analyzing the studies which have focused on diverse TCP techniques using the criteria of PO's framed for this study and results are thus documented in above table.

5. CONCLUSION AND FUTURE SCOPE

Test Case Prioritization (TCP) is one of the approaches of regression testing which aims to make testing effective and efficient. This paper attempts to cover almost all TCP techniques proposed by numerous researchers. The prioritization objectives (PO) are framed for this study which many researchers have kept in mind while developing these techniques. Based on these PO, selected studies have been compared to find which particular approach fits in our desired criteria. It can be concluded that every prioritization technique has its advantages and limitations which are according to the testing needs, size of program, requirements and testing environment. So, it is not appropriate to say that this particular technique has this much of flaws or these advantages as it is based on the number of factors discussed above. Every researcher wants to develop a cost-effective technique, which may find faults early and is fast in execution also. Every aspect cannot be assured in a particular technique, though researchers will try to do so. For it, careful selection is required to be done keeping in mind various factors. It is not feasible to apply every existing TCP technique on different test suites/testing phases/software product to find the best TCP technique. If one technique is selected from them, then it may or may not be suitable for a different software product/test-suite/testing phase. Further study in this direction will try to perform an experimental analysis to compare the results obtained from the software product/ test suite when different TCP techniques are applied to it.

REFERENCES

1. A. Mathur, ***"Foundation of software testing"***, second edition, Pearson publications, 2013.
2. J. Bentley, ***"Software testing fundamentals–concepts, roles, and terminology",*** In *SUGI*, volume 30, 2005.
3. E. Dijkstra, **"On the reliability of mechanisms",** *Notes on Structured Programming*, 1970.
4. S. Desikan and G. Ramesh, ***"Software testing principles & practices"***, Pearson Education, 2007.
5. R. S. Pressman**, *"Software engineering: a practitioner's approach",*** Palgrave Macmillan, 2005.
6. K. Solanki, and Y. Singh, **"Importance of Selecting Test Cases for Regression Testing",** *IOSR Journal of Computer Engineering (IOSRJCE) e-ISSN*, pp. 2278-0661, 2014. https://doi.org/10.9790/0661-16444351
7. K. Solanki, and Y. Singh, **"Novel Classification of Test Case Prioritization Techniques",** *International Journal of Computer Applications*, Vol. 975, pp. 8887, 2014.
8. R. Ramler and K. Wolfmaier, **"Economic perspectives in test automation: balancing automated and manual testing with opportunity cost",** In *Proceedings of the 2006 international workshop on Automation of software test* ACM, pp. 85-91, May 2006. https://doi.org/10.1145/1138929.1138946
9. **https://time.com/5615292/boeing-737-max-software-problem/.**
10. **https://www.wired.co.uk/article/what-is-eternal-blue-exploit-vulnerability-patch.**
11. G. Rothermel and M.J. Harrold, **"Empirical studies of a safe regression test selection technique",** *IEEE Transactions on Software Engineering*, vol. *24* No. 6, pp. 401-419, 1998. https://doi.org/10.1109/32.689399
12. G. Rothermel, and J. Harrold, **"Analyzing regression test selection techniques"**. *IEEE Transactions on Software Engineering*, Vol. 22, No. 8 pp. 529-551, Aug. 1996. https://doi.org/10.1109/32.536955
13. A. Orso, T. Apiwattanapong and M. J. Harrold, **"Leveraging Field Data for Impact Analysis and Regression Testing,"** In *ACM SIGSOFT Software Engineering Notes*, vol. 28, no. 5, pp. 128-137. ACM, 2003. https://doi.org/10.1145/949952.940089
14. B. Beizer, ***"Software testing techniques"***, 2nd Edition, Van Nostrand Reinhold, New York, 1990.
15. O. Dahiya and K. Solanki, **"A systematic literature study of regression test case prioritization approaches".** *International Journal of Engineering & Technology*, Vol. 7, No. 4, pp.2184-2191, 2018. https://doi.org/10.14419/ijet.v7i4.15805
16. S. Yoo and M. Harman, **"Regression Testing Minimisation, Selection, and Prioritization: A survey,"** *Journal of software testing, Verification, and Reliability*, vol. 22, no. 2, pp. 67-120, 2012. https://doi.org/10.1002/stv.430
17. S. Elbaum, P. Kallakuri, A. Malishevsky, G. Rothermel, and S. Kanduri, **"Understanding the effects of changes on the cost□effectiveness of regression testing techniques".** *Software testing, verification, and reliability*, Vol. *13* No. 2, pp.65-83, April 2003. https://doi.org/10.1002/stvr.263
18. G. Rothermel, M. J. Harrold, J. Ostrin, and C. Hong, 1998, November. **"An empirical study of the effects of minimization on the fault detection capabilities of test suites".** In *Proceedings. International Conference on Software Maintenance (Cat. No. 98CB36272)*, *IEEE*, pp. 34-43, Nov. 1998.
19. R. Mukherjee and K. S. Patnaik, **"A survey on different approaches for software test case prioritization",** *Journal of King Saud University-Computer and Information Sciences*, 2018. https://doi.org/10.1016/j.jksuci.2018.09.005
20. A. Kumar, & K. Singh, **"A Literature Survey on test case prioritization".** *Compusoft*, Vol. *3*, No. 5, p. 793, May 2014.
21. R. S. Kiran, **"A literature survey on TCP-test case prioritization using the RT-regression techniques".** *Global Journal of Research in Engineering*, April 2015.
22. C. Catal, & D. Mishra, **"Test case prioritization: a systematic mapping study".** *Software Quality Journal*, Vol. *21*, No. 3, pp. 445-478, Sept. 2013. https://doi.org/10.1007/s11219-012-9181-z
23. Y. Singh, A. Kaur, B. Suri, & S. Singhal, **"Systematic literature review on regression test prioritization techniques".** *Informatica*, Vol. *36*, No. 4, 2012.
24. M. Khatibsyarbini, M. A. Isa, D. N. Jawawi and R. Tumeng, **"Test case prioritization approaches in regression testing: A systematic literature review".** *Information and Software Technology*, *93*, pp.74-93, 2018. https://doi.org/10.1016/j.infsof.2017.08.014
25. B. Kitchenham, **"Procedures for performing systematic reviews".** *Keele, UK, Keele University*, Vol. *33*, No. 2004, pp. 1-26, 2004.
26. Y. Lou, D. Hao, and L. Zhang, **"Mutation-based test-case prioritization in software evolution",** *IEEE 26th International Symposium on Software Reliability Engineering, ISSRE*, pp. 46–57, 2015. https://doi.org/10.1109/ISSRE.2015.7381798
27. R. Maheswari and D. Mala, **"Combined Genetic and Simulated Annealing Approach for Test CasePrioritization,"** *Indian Journal of Science and Technology*, Vol. 8, No. 35, p. 1, 2015. https://doi.org/10.17485/ijst/2015/v8i35/81102
28. C. Catal, **"On the application of genetic algorithms for test case prioritization: a systematic literature**

review". In *Proceedings of the 2nd international workshop on evidential assessment of software technologies*, ACM, pp. 9-14, Sept. 2012. https://doi.org/10.1145/2372233.2372238

29. F. Yuan, Y. Bian, Z. Li, and R. Zhao, **"Epistatic Genetic Algorithm for Test Case Prioritization**,**"** *International Symposium on Search Based*, pp. 109-124, Sept. 2015. https://doi.org/10.1007/978-3-319-22183-0_8
30. K. Deb, S. Pratab, S. Agarwal, and T. Meyarivan, **"A Fast and Elitist Multiobjective Genetic Algorithm: NGSA-II**,**"** *IEEE Transactions on Evolutionary Computing*, vol. 6, no. 2, pp. 182–197, Aug. 2002. https://doi.org/10.1109/4235.996017
31. A. Kaur and S. Goyal, **"A genetic algorithm for fault-based regression test case prioritization**,**"** *International Journal of Computer Applications*, vol. 32, no. 8, pp. 975–8887, Oct. 2011.
32. W. Jun, Z. Yan, and J. Chen, **"Test case prioritization technique based on genetic algorithm**,**"** *Internet Computing & Information*, pp. 173-175, Sept. 2011.
33. S. Sabharwal, R. Sibal, and C. Sharma, **"Prioritization of test case scenarios derived from activity diagram using genetic algorithm",** *2010 International Conference on Computer and Communication Technology, ICCCT-2010, IEEE*, pp. 481–485, 2010.
34. Y. C. Huang, C. Y. Huang, J. R. Chang, and T. Y. Chen, **"Design and analysis of cost cognizant test case prioritization using genetic algorithm with test history**,**"** In *2010 IEEE 34th Annual Computer Software and Applications Conference*, pp. 413–418, July 2010.
35. Y. Singh, A. Kaur, and B. Suri, **"Test case prioritization using ant colony optimization**,**"** *ACM SIGSOFT Softw. Eng. Notes*, vol. 35, no. 4, pp. 1-7, 2010
36. D. Gao, X. Guo, and L. Zhao, **"Test case prioritization for regression testing based on ant colony optimization",** *In 2015 6th IEEE International Conference on Software Engineering and Service Science (ICSESS), IEEE*, pp. 275-279. *2015***.**
37. B. Suri, & S. Singhal, **"Analyzing test case selection & prioritization using ACO"**. *ACM SIGSOFT Software Engineering Notes*, Vol. *36*, No. 6, pp. 1-5, 2011.
38. T. Noguchi, H. Washizaki, and Y. Fukazawa, **"History-Based Test Case Prioritization for Black Box Testing Using Ant Colony Optimization**,**"** In *2015 IEEE 8th International Conference on Software Testing, Verification and Validation (ICST), IEEE*, pp. 1-2, 2015.
39. K. Solanki, Y. Singh, S. Dalal, and P.R. Srivastava, **"Test case prioritization: an approach based on modified ant colony optimization".** In *Emerging Research in Computing, Information, Communication and Applications*, Springer, Singapore, pp. 213-223, 2016.
40. S. Eghbali and L. Tahvildari, **"Test Case Prioritization Using Lexicographical Ordering**,**"** *IEEE Transactions on Software Engineering*, Vol. 42, No.12, pp.1178-1195, April 2016.
41. D. J. Mala, M. Kamalapriya, R. Shobana, and V. Mohan, **"A non-pheromone based intelligent swarm optimization technique in software test suite optimization",** *In 2009 International Conference on Intelligent Agent & Multi-Agent Systems, IEEE*, pp. 1-5. July 2009.
42. D. Jeffrey and N. Gupta, **"Test case prioritization using relevant slices",** In *30th Annual International Computer Software and Applications Conference (COMPSAC'06), IEEE*, Vol. 1, pp. 411-420, Sept. 2006.
43. S. Li, N. Bian, Z. Chen, and D. You, **"A simulation study on some search algorithms for regression test case prioritization"** *In 2010 10th International Conference on Quality Software, IEEE*, pp. 72-81, 2010.
44. D. Nardo, N. Alshahwan, and L. Briand, **"Coverage□ based regression test case selection, minimization, and prioritization: a case study on an industrial system",** *Software Testing, Verification and Reliability*, Vol. 25, No. 4, pp.371-396, 2015.
45. H. Do and G. Rothermel, **"On the use of mutation faults in empirical assessments of test case prioritization techniques",** *IEEE Transactions on Software Engineering*, Vol. 32, No. 9, pp. 733-752, 2006.
46. H. Do, G. Rothermel, and A. Kinneer, **"Prioritizing JUnit test cases: An empirical assessment and cost-benefits analysis",** *Empirical Software Engineering*, Vol. 11, No. 1, pp. 33-70, 2006.
47. G. Rothermel, R. H. Untch, C. Chu and M. J. Harrold, **"Prioritizing test cases for regression testing"**. *IEEE Transactions on software engineering*, Vol. 27, No. 10, pp.929-948, 2001.
48. G. M. Kapfhammer and M. L. Soffa, **"Using coverage effectiveness to evaluate test suite prioritizations",** *In Proceedings of the 1st ACM international workshop on Empirical assessment of software engineering languages and technologies: held in conjunction with the 22nd IEEE/ACM International Conference on Automated Software Engineering (ASE)*, ACM, pp. 19-20, Nov. 2007.
49. D. Hao, L. Zhang, L. Zang, Y. Wang, X. Wu, T. Xie, **"To Be Optimal or Not in Test-Case Prioritization"**. *IEEE Transactions on Software Engineering*, Vol. 42, No. 5, pp. 490-504, 2015.
50. C. Fang, Z. Chen, and B. Xu, **"Comparing logic coverage criteria on test case prioritization",** *Science China Information Sciences*, Vol. 55, No. 12, pp.2826-2840, 2012.
51. S. Elbaum, A. G. Malishevsky, and G. Rothermel, **"Test case prioritization: A family of empirical studies".** *IEEE transactions on software engineering*, Vol. 28, No. 2, pp.159-182, 2002.
52. D. Di Nardo, N. Alshahwan, L. Briand, and Y. Labiche, **"Coverage-based test case prioritisation: An industrial case study",** In *2013 IEEE Sixth*

International Conference on Software Testing, Verification and Validation, IEEE, pp. 302-311, 2013. https://doi.org/10.1109/ICST.2013.27

53. J. M. Kim, and A. Porter, **"A history-based test prioritization technique for regression testing in resource constrained environments",** In *Proceedings of the 24th international conference on software engineering* ACM, pp. 119-129, May 2002.
54. A. Khalilian, M. A. Azgomi, and Y. Fazlalizadeh, **"An improved method for test case prioritization by incorporating historical test case data"**, *Science of Computer Programming*, Vol. 78. No. 1, pp.93-116, Nov. 2012.
55. H. Park, H. Ryu and J. Baik, **"Historical value-based approach for cost-cognizant test case prioritization to improve the effectiveness of regression testing"**, In *2008 Second International Conference on Secure System Integration and Reliability Improvement*, IEEE, pp. 39-46, July 2008.
56. S. Kim and J. Baik, **"An effective fault aware test case prioritization by incorporating a fault localization technique"**, In *Proceedings of the 2010 ACM-IEEE International Symposium on Empirical Software Engineering and Measurement*, ACM, p. 5, Sept. 2010.
57. Y. Fazlalizadeh, A. Khalilian, M. A. Azgomi and S. Parsa, **"Prioritizing test cases for resource constraint environments using historical test case performance data",** In *2009 2nd IEEE International Conference on Computer Science and Information Technology*, *IEEE*, pp. 190-195, Aug. 2009.
58. H. Aman, T. Nakano, H. Ogasawara and M. Kawahara, **"A topic model and test history-based test case recommendation method for regression testing",** In *2018 IEEE International Conference on Software Testing, Verification and Validation Workshops (ICSTW), IEEE*, pp. 392-397, April 2018.
59. T. Megala and K. Vivekanadan, **"History Based Multi Objective Test Suite Prioritization in Regression Testing Using Genetic Algorithm",** pp. 129–135, 2017.
60. H. Srikanth, L. Williams and J. Osborne, **"System test case prioritization of new and regression test cases"**. In *2005 International Symposium on Empirical Software Engineering, IEEE, pp.* 10-pp, Nov. 2005.
61. T. Muthusamy, **"A New Effective Test Case Prioritization for Regression Testing based on Prioritization Algorithm",** *International Journal of Applied Information Systems (IJAIS)*, vol. 6, no. 7, pp. 21–26, 2014.
62. H. Srikanth, C. Hettiarachchi, and H. Do, **"Requirements Based Test Prioritization Using Risk Factors",** *Information and Software Technology*, vol. 69, no. C, pp. 71–83, 2016.
63. H. Srikanth and L. Williams, **"On the economics of requirements-based test case prioritization"**, In *ACM SIGSOFT Software Engineering Notes*, ACM, Vol. 30, No. 4, pp. 1-3, May 2005.
64. H. Srikanth, S. Banerjee, L. Williams and J. Osborne, **"Towards the prioritization of system test cases", Software** *Testing, Verification and Reliability*, Vol. 24, No. 4, pp.320-337, June 2014. https://doi.org/10.1002/stvr.1500
65. T. Ma, H. Zeng, and X. Wang, **"Test case prioritization based on requirement correlations"**, *IEEE/ACIS 17th International Conference on Software Engineering, Artificial Intelligence, Networking, and Parallel/Distributed Computing, SNPD*, pp. 419–424, May 2016.
66. M. J. Arafeen and H. Do, **"Test case prioritization using requirements-based clustering"** In *2013 IEEE Sixth International Conference on Software Testing, Verification and Validation, IEEE, pp.* 312-321, 2013.
67. R. Krishnamoorthi and S.S.A Mary, **"Factor oriented requirement coverage-based system test case prioritization of new and regression test cases",** *Information and Software Technology*, Vol. 51, No. 4, pp.799-808, April 2009.
68. X. Zhang, C. Nie, B. Xu and B. Qu, 2007, **"Test case prioritization based on varying testing requirement priorities and test case costs",** In *Seventh International Conference on Quality Software (QSIC 2007), IEEE*, pp. 15-24, Oct. 2007.
69. K. Ramasamy and S. A. Mary, 2008, **"Incorporating varying requirement priorities and costs in test case prioritization for new and regression testing"**, In *2008 International Conference on Computing, Communication and Networking, IEEE*, pp. 1-9, 2008.
70. Y. I. Salem and R. Hassan, **"Requirement-based test case generation and prioritization"**, In *2010 International Computer Engineering Conference (ICENCO), IEEE*, pp. 152-157, Dec. 2010.
71. R. Kavitha, V. R. Kavitha and N. S. Kumar, **"Requirement based test case prioritization"**, In *2010 International Conference on Communication Control and Computing Technologies, IEEE*, pp. 826-829, 2010.
72. P. Berander and A. Andrews, "**Requirements Prioritization"**, in *Engineering and Managing Software Requirements*, A. Aurum and C. Wohlin, Eds. Berlin, Heidelberg: Springer Berlin Heidelberg, pp. 69–94, 2005.
73. M. Salehie, S. Li, L. Tahvildari, R. Dara, S. Li and M. Moore, **"Prioritizing requirements-based regression test cases**: **A goal-driven practice",** In *2011 15th European Conference on Software Maintenance and Reengineering, IEEE, pp.* 329-332, March 2011. https://doi.org/10.1109/CSMR.2011.46
74. S. P. Dongoor, **"Selecting an appropriate Requirements Based Test Case Prioritization Technique",** M.S. dissertation, 2019.
75. G. Rothermel, R. H. Untch, C. C. Chu, and M. J. Harrold, "**Test case prioritization: an empirical study"**, in *Software Maintenance, 1999. (ICSM '99) Proceedings. IEEE International Conference on Software Maintenance-1999 (ICSM'99).'Software*

Maintenance for Business Change'(Cat. No. 99CB36360), IEEE, pp. 179–188, Aug. 1999.

76. Y. T. Yu and M. F. Lau, **"Fault-based test suite prioritization for specification-based testing"** *Information and Software Technology*, vol. 54, no. 2, pp. 179–202, 2012.
77. K. Solanki, Y. Singh and S. Dalal, **"A Comparative Evaluation of "m-ACO" Technique for Test Suite Prioritization"**, *Indian Journal of science and technology*, Vol. 9, No. 30, pp.1-10, Aug. 2016.
78. B. Jiang, Z. Zhang, W. K. Chan, T. H. Tse, and T. Y. Chen, **"How well does test case prioritization integrate with statistical fault localization?"**, *Information and Software Technology*, vol. 54, no. 7, pp. 739–758, July 2012.
79. K. Solanki, Y. Singh, and S. Dalal, **"Test case prioritization: an approach based on modified ant colony optimization (m-ACO)"**, In *2015 International Conference on Computer, Communication and Control (IC4), IEEE*, pp. 1-6, Sept. 2015.
80. Y. C. Huang, K. L. Peng, and C. Y. Huang, **"A history-based cost-cognizant test case prioritization technique in regression testing"**, *Journal of Systems and Software*, vol. 85, no. 3, pp. 626–637, March 2012.
81. A. G. Malishevsky, J. R. Ruthruff, G. Rothermel and S. Elbaum, **"Cost-cognizant test case prioritization",** *Technical Report TR-UNL-CSE-2006-0004, University of Nebraska-Lincoln*, (pp. 97-106), March 2006.
82. A. G. Malishevsky, G. Rothermel, and S. Elbaum, **"Modeling the cost-benefits tradeoffs for regression testing techniques"** In *International Conference on Software Maintenance, 2002. Proceedings, IEEE, pp.* 204-213, Oct. 2002.
83. S. Elbaum, G. Rothermel, S. Kanduri, S. and A. G. Malishevsky, **"Selecting a cost-effective test case prioritization technique"**, *Software Quality Journal*, Vol. 12, No. 3, pp.185-210, Sept. 2004.
84. A.M. Smith and G.M Kapfhammer, **"An empirical study of incorporating cost into test suite reduction and prioritization"**, In *Proceedings of the 2009 ACM symposium on Applied Computing, ACM, pp.* 461-467, March 2009.
85. H. Srikanth, M. B. Cohen and X. Qu, 2009, **"Reducing field failures in system configurable software: Cost-based prioritization"**, In *2009 20th International Symposium on Software Reliability Engineering, IEEE*, pp. 61-70, Nov. 2009.
86. P.R Srivastava, **"Model for optimizing software testing period using non homogenous poisson process based on cumulative test case prioritization",** In *TENCON 2008-2008 IEEE Region 10 Conference, IEEE*, pp. 1-6, Nov. 2008.
87. H. Stallbaum, A. Metzger and K. Pohl, **"An automated technique for risk-based test case generation and prioritization",** In *Proceedings of the 3rd international workshop on Automation of software test*, ACM, pp. 67-70, May 2008.
88. M. Yoon, E. Lee, M. Song and B. Choi, **"A test case prioritization through correlation of requirement and risk",** *Journal of Software Engineering and Applications*, Vol. 5, No. 10, p.823, Oct. 2012. https://doi.org/10.4236/jsea.2012.510095
89. C. Hettiarachchi, H. Do, and B. Choi, **"Effective regression testing using requirements and risks",** *Proceedings - 8th International Conference on Software Security and Reliability (SERE), IEEE*, pp. 157–166, June 2014.
90. H. Yoon and B. Choi, **"**A **Test Case Prioritization Based on Degree of Risk Exposure and Its Empirical Study",** *International Journal of Software Engineering and Knowledge Engineering*, vol. 21, no. 2, pp. 191–209, 2011.
91. C. Hettiarachchi, H. Do, and B. Choi, **"Risk-based test case prioritization using a fuzzy expert system",** *Information and Software Technology*, 69, pp. 1-15, Jan. 2016.
92. E. J. Uusitalo, M. Komssi, M. Kauppinen and A. M. Davis, **"Linking requirements and testing in practice",** In *2008 16th IEEE International Requirements Engineering Conference, IEEE*, pp. 265-270, Sept. 2008.
93. G. Mogyorodi, **"Requirements-based testing: an overview",** *In Proceedings 39th International Conference and Exhibition on Technology of Object-Oriented Languages and Systems. TOOLS 39, IEEE, pp.* 286-295, July 2001.
94. S, W. Thomas, H. Hemmati, A. E. Hassan and D. Blostein, **"Static test case prioritization using topic models",** *Empirical Software Engineering*, Vol. 19, No. 1, pp. 182-212, Feb. 2014.
95. B. Korel, L. H. Tahat and M. Harman, **"Test prioritization using system models",** In *21st IEEE International Conference on Software Maintenance (ICSM'05), IEEE*, pp. 559-568, Sept. 2005.
96. B. Korel, G. Koutsogiannakis and L. H. Tahat, **"Model-based test prioritization heuristic methods and their evaluation",** In *Proceedings of the 3rd international workshop on Advances in model-based testing*, ACM, pp. 34-43, July 2007.
97. B. Korel, G. Koutsogiannakis and L. H. Tahat, **"Application of system models in regression test suite prioritization",** In *2008 IEEE International Conference on Software Maintenance*, IEEE, pp. 247-256, 2008.
98. H. Hemmati, A. Arcuri and L. Briand, **"Achieving scalable model-based testing through test case diversity",** *ACM Transactions on Software Engineering and Methodology (TOSEM)*, Vol. 22, No. 1, p.6, 2013.
99. C. R. Panigrahi and R. Mall, **"A heuristic-based regression test case prioritization approach for object-oriented programs",** *Innovations in Systems and Software Engineering*, Vol. 10, No. 3, pp.155-163, 2014.

100. C. R. Panigrahi and R. Mall, **"Model-based regression test case prioritization",** *ACM SIGSOFT Software Engineering Notes*, Vol. 35, No. 6, pp.1-7, 2010. https://doi.org/10.1145/1874391.1874405
101. S. Sampath, R.C. Bryce, G. Viswanath, V. Kandimalla, V. and A.G. Koru, **"Prioritizing user-session-based test cases for web applications testing",** In *2008 1st International Conference on Software Testing, Verification, and Validation, IEEE*, pp. 141-150, 2008.
102. A. M. Memon and Q. Xie, **"Studying the fault-detection effectiveness of GUI test cases for rapidly evolving software",** *IEEE transactions on software engineering*, Vol. 31, No. 10, pp.884-896, Nov. 2005.
103. S. Sprenkle, S. Sampath, E. Gibson, L. Pollock and A. Souter, **"An empirical comparison of test suite reduction techniques for user-session-based testing of web applications",** In *21st IEEE International Conference on Software Maintenance (ICSM'05), IEEE*, pp. 587-596, Sept. 2005.
104. R. C. Bryce and A. M. Memon, **"Test suite prioritization by interaction coverage",** In *Workshop on Domain specific approaches to software test automation: in conjunction with the 6th ESEC/FSE joint meeting, ACM*, pp. 1-7, Sept. 2007.
105. R. C. Bryce, S. Sampath and A. M. Memon, **"Developing a single model and test prioritization strategies for event-driven software",** *IEEE Transactions on Software Engineering*, Vol. 37, No. 1, pp.48-64, 2010.
106. K.H.S. Hla, Y. Choi, and J.S Park, **"Applying particle swarm optimization to prioritizing test cases for embedded real time software retesting",** In *2008 IEEE 8th International Conference on Computer and Information Technology Workshops, IEEE*, pp. 527-532, July 2008.
107. A. Srivastava and J. Thiagarajan, **"Effectively prioritizing tests in development environment",** In *ACM SIGSOFT Software Engineering Notes*, ACM, Vol. 27, No. 4, pp. 97-106, July 2002.
108. Y. Lu, Y. Lou, S. Cheng, L. Zhang, D. Hao, Y. Zhou and L. Zhang, **"How does regression test prioritization perform in real-world software evolution?",** In *2016 IEEE/ACM 38th International Conference on Software Engineering (ICSE), IEEE*, pp. 535-546, May 2016.
109. A. Vescan, C. Şerban, C. Chisăliţă-Cretu, and L. Dioşan, **"Requirement dependencies-based formal approach for test case prioritization in regression testing",** In *2017 13th IEEE International Conference on Intelligent Computer Communication and Processing (ICCP), IEEE*, pp. 181-188, Sept. 2017.
110. A. Schwartz and H. Do, **"Cost-effective regression testing through Adaptive Test Prioritization strategies",** *Journal of Systems and Software, 115*, pp.61-81, May 2016. https://doi.org/10.1016/j.jss.2016.01.018
111. Sujata and G. N. Purohit, 2015, **"A Schema Support for Selection of Test Case Prioritization Techniques",** In *2015 Fifth International Conference on Advanced Computing & Communication Technologies, IEEE*, pp. 547-551, Feb. 2015. https://doi.org/10.1109/ACCT.2015.91