iJRASET
International Journal For Research in
Applied Science and Engineering Technology

INTERNATIONAL JOURNAL FOR RESEARCH
IN APPLIED SCIENCE & ENGINEERING TECHNOLOGY

Volume: 6 Issue: IV Month of publication: April 2018
DOI: http://doi.org/10.22214/ijraset.2018.4671

www.ijraset.com
Call: 08813907089 | E-mail ID: ijraset@gmail.com

A Systematic Analysis and Exploration of the Regression Testing Techniques

Ramandeep Kaur[1], Manpreet Singh[2], Rishideep Singh[3]
[1]Research Scholar, [2, 3]Assistant Professor Department of Computer Science & Engineering, North West Institute Of Engineering &Technology, Dhudike, Moga

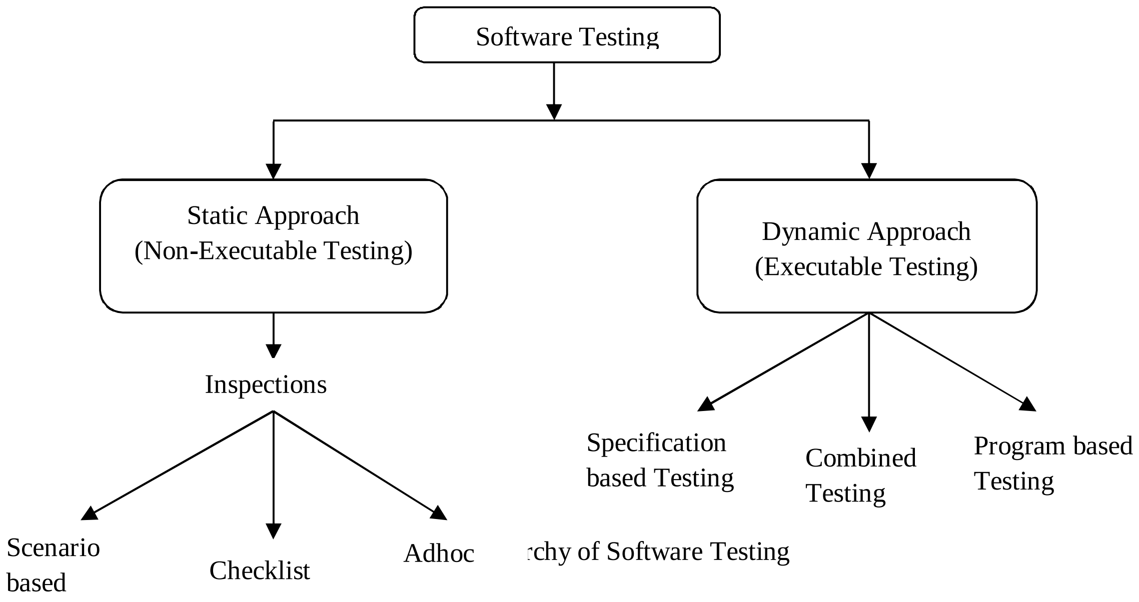
Abstract: Software system undergoes various evolutions from its initial development to practical use. There are various kinds of corrections, modifications and additions at each module in the software. Sometimes, software is shifted from one platform to another platform which can adversely affect the software system. To ensure the system from bugs, faults & errors, the system undergoes the regression testing. Regression testing ensures the quality of the software with testing all the required phases. It is easy to perform regression testing to ensure the proper working of software in case small software or software having limited programming code lines but it becomes difficult after the addition of phases and new modules to software. It also becomes costlier to perform regression testing for the large projects having large number of test cases. To resolve this problem, there are various techniques of regression testing like test suite minimization, prioritization of test cases and test case selection. Researchers are working on these techniques to handle large projects by using different methods on mentioned regression testing techniques. In this paper, we have explored the regression testing techniques and analyzed new methods and trends in the field of regression testing. The presented analysis indicates the continuous growth of the concept in wider research applications.
Keywords: Software Development, Software Testing, Regression Testing, Software Quality Improvement, Test Case Selection, Test Case Prioritization, Test Suite Minimization

I. INTRODUCTION

Software testing works as a key to ensure the software quality. The program developed by developers is doubtful regarding the correctness and performance until the software testing ensures the quality of software. Correctness and quality of software product indicates the best performance iff the suitable results can be achieved in all the relevant domains. However, testing process may contain some chances of error but it ensures the quality upto the maximum possible extent. So, testing process play an important role in the software development life cycle. Early detection of faults can reduce the cost. Therefore, there is always need to update the testing technique to generate optimum test cases. There are two methods of testing such as static and dynamic. Static approach performs the software testing without any code execution and dynamic approach performs the testing with code execution. Static approach is a verification process and dynamic technique is validation approach. The complete hierarchy of software testing [1] is presented in fig. 1. In fig. 1, there are further three types of dynamic approach such as specification based testing, combined testing and program based testing. Specification based testing handles the error based criterion. Program based testing handles the structurally based criterion and combined testing handles the fault based criterion. On the other hand, the inspection of static approach can be performed as scenario based, checklist and adhoc.

Dynamic (execution based testing) is used in this research work as regression testing is performed while execution of code. The complete process of dynamic testing involves the elements of program code, test case specification, test case generation, execution, regression testing and test results. This complete process is presented in fig. 2.

Regression testing can be defined as the process to test the modified software to check and ensure its functionality [2]. Regression testing becomes the part of software development life cycle. It is also universally employed module to ensure the software quality but it becomes costlier in case of large software projects. There are various instances like one software product contains regression based 30,000 test cases that directly demand for 1000 hours for the execution on machine. This also need thousands of hours to overlook the process to maintain testing resources, monitor results and set up the test cases. So, overall there is need to optimize the regression testing. From the existing four techniques of regression testing (Retest entire approach, Test Suite Minimization, Regression Test Case Selection and Test Case Prioritization), later three mentioned techniques can be used to reduce the cost of testing. The basic concepts of these four techniques are discussed in section 2. There are various researchers working on these techniques to optimize the regression testing and provide the better quality with lower software cost.


Software Testing
Static Approach (Non-Executable Testing)
Dynamic Approach (Executable Testing)
Inspections
Scenario based
Checklist
Adhoc
Specification based Testing
Combined Testing
Program based Testing

rchy of Software Testing

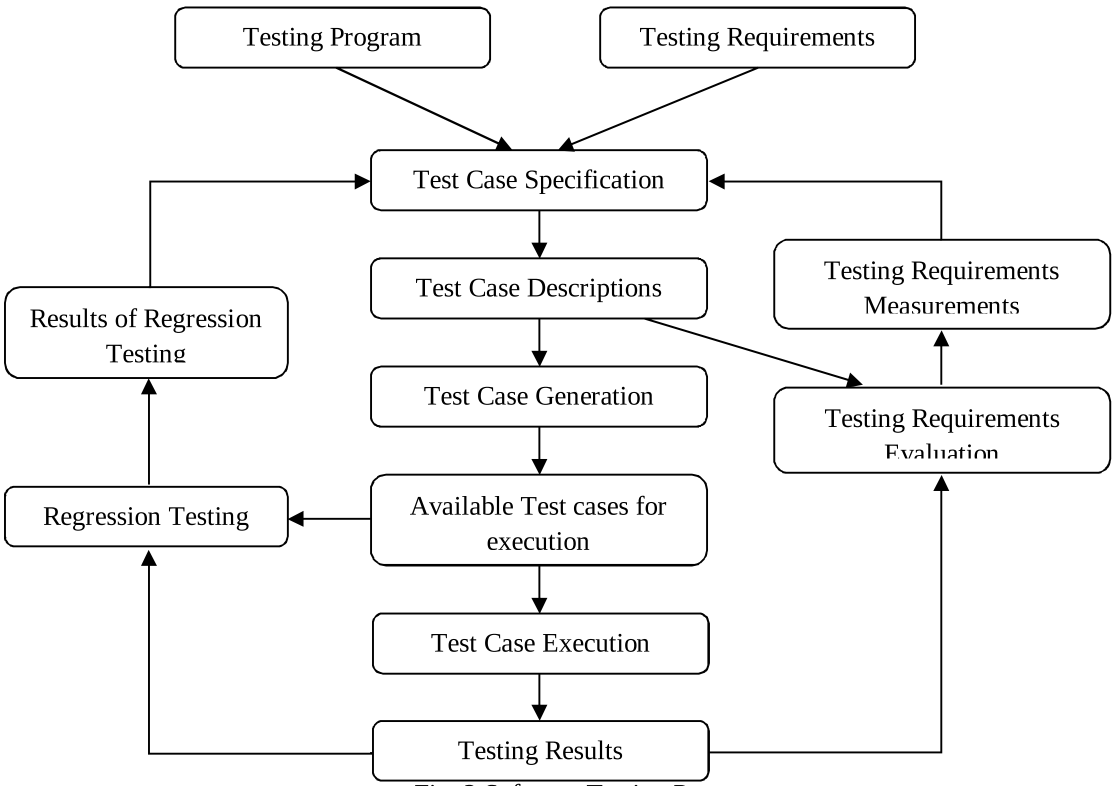


Fig. 2 Software Testing Process

In this paper, we have presented an analytical review on the existing methods for the regression testing. Different authors have used the regression testing by using different concept. Also the basic techniques of regression testing are discussed. Finally, an overall evaluation from the existing concepts is presented.

Dynamic (execution based testing) is used in this research work as regression testing is performed while execution of code. The complete process of dynamic testing involves the elements of program code, test case specification, test case generation, execution, regression testing and test results. This complete process is presented in fig. 2.

Regression testing can be defined as the process to test the modified software to check and ensure its functionality [2]. Regression testing becomes the part of software development life cycle. It is also universally employed module to ensure the software quality but it becomes costlier in case of large software projects. There are various instances like one software product contains regression based 30,000 test cases that directly demand for 1000 hours for the execution on machine. This also need thousands of hours to overlook the process to maintain testing resources, monitor results and set up the test cases. So, overall there is need to optimize the regression testing. From the existing four techniques of regression testing (Retest entire approach, Test Suite Minimization, Regression Test Case Selection and Test Case Prioritization), later three mentioned techniques can be used to reduce the cost of testing. The basic concepts of these four techniques are discussed in section 2. There are various researchers working on these techniques to optimize the regression testing and provide the better quality with lower software cost.

In this paper, we have presented an analytical review on the existing methods for the regression testing. Different authors have used the regression testing by using different concept. Also the basic techniques of regression testing are discussed. Finally, an overall evaluation from the existing concepts is presented.

This section covers the basic introduction of software testing, its processing model and basic of regression testing. Section 2 presents the basic four techniques of regression testing. Section 3 presents the existing literature work on regression testing. Section 4 presents the analysis and observations from the existing techniques. Section 5 concludes the paper with some future directions.

II. REGRESSION TESTING TECHNIQUES

This section presents the basic techniques of regression testing. These techniques are Retest entire approach, test suite minimization, regression test case selection, and test case prioritization. These techniques are presented in fig. 3 and discussed further.

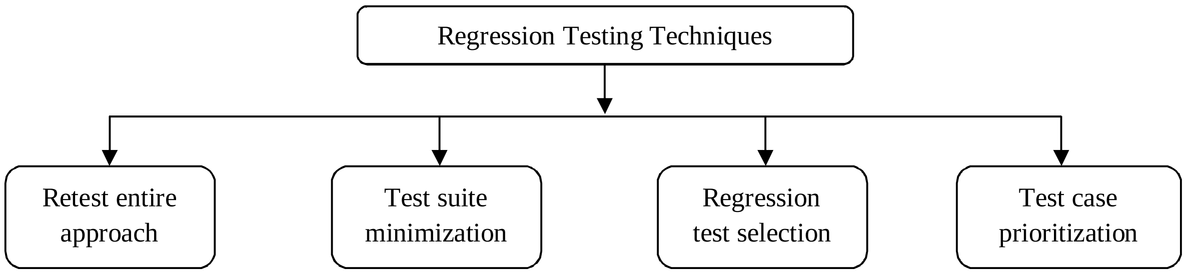


Fig. 3 Regression Testing Techniques

III. LITERATURE REVIEW ON EXISTING METHODS

In this section, a review on the existing regression testing techniques is presented. Techniques is Reset entire approach is the concept to retest all the modules of the software which is not considered by much authors. So, most of the literature review work is presented by focusing on test suite minimization, regression test case selection, and test case prioritization.

A. *Test Suite Minimization*

The approach of test suite minimization focuses on the reduction of test cases available in test suite without any impact on the software quality assurance. Selvakumar et al. [3] have used particle swarm optimization for the test case minimization along with the concept analysis method for the effectiveness of fault detection. Ahmed et al. [4] have used Cuckoo Search (CS) approach for the test suite minimization and evaluated results with variation in the parameters of CS algorithm. Huang et al. [5] have proposed the concept of fuzzy expert system (FES) with traditional approaches of Greedy algorithm, GRE and Harrold-Gupta-Soffa (HGS) approach. Concept is experimented on nine real time programming codes. Results are evaluated in terms of execution time, fault detection capability etc. Zhang et al, [6] have modified the concept of ant colony optimization with quantum inspired evolutionary approach and quantum gate to reduce the test suite cases with different simulation results. But authors have observed the interruption in local search and slower coverage rate. Liu et al. [7] have used K-medoids clustering approach after observing the drawbacks of k-means clustering algorithm. Further, Indumathi and Madhumathi [8] have used Maximum Frequent Test set for the test suite reduction and genetic algorithm for the test case prioritization. Results are observed in terms of Average Percentage of Fault Detection metric (APFD). Singhal et al. [9] have proposed MHBG_TCS approach for the test suite minimization. MHBG_TCS approach is hybrid of bee colony optimization and genetic algorithm with focus on time constraint in regression testing. Yamuc et al. [10] have used greedy algorithm and genetic algorithm for the test suite minimization. Authors have achieved efficient performance of the proposed approach for suite reduction. Kabir et al. [11] have used modified flower pollination algorithm for the test suite reduction. Here, flower pollination algorithm is modified by updation of step length parameter at each iteration. Further, Sugave et al. [12] have proposed Diversity Dragonfly Algorithm (DDF) for the test suite reduction with the diversification of three bitwise operators in standard dragonfly algorithm. Table I presents the summarized form of considered existing literature review concepts on test suite minimization.

TABLE I EXISTING CONCEPTS OF TEST SUITE MINIMIZATION

Author and Year	Used Method	Remarks
Selvakumar et al., 2013	Conceptual Particle Swarm Optimization (CPSO), Genetic algorithm (GA)	• Concept analysis approach is used for the fault analysis along with PSO for test suite minimization. • Results are evaluated for fault detection capability with reduction on test suite with genetic algorithm in some cases. • Observed Inferior performance of PSO than GA in some cases.
Ahmed et al., 2015	Cuckoo Search (CS)	• Observed efficient results for the combinatorial test suite minimization with different parametric variation in CS approach
Huang et al. 2016	FES-Greedy, FES-GRE, FES-HGS	• Reported better fault detection capability of FES-GRE approach in comparison with GRE and other existing concepts. • FES system is not suitable for non-numeric inputs
Zhang et al, 2017	Modified Quantum Ant Colony Optimization	• Considered the concept as single objective problem • Reported slower coverage extent and trapping of ACO in local search. • Suggested to combine with other optimization algorithms
Liu et al., 2017	K-medoids clustering approach, Greedy algorithm	• Authors have used greedy algorithm in the large coverage based test suite screening • Final small test suite is selected based on k-medoid approach
Indumathi & Madhumathi, 2017	Genetic algorithm, Maximum Frequent Test set (MFTS)	• Results are evaluated based on APFD metric for both the prioritized & non-prioritized test cases and observed better test suite reduction results with prioritized test cases.
Singhal et al., 2017	MHBG_TCS	• Achieved possible reduction in test suite. • Results are evaluated with different suite size, execution time and possible optimal suite reduction.
Yamuc et al., 2017	Greedy Algorithm, Genetic algorithm	• Genetic algorithm is considered to overcome the drawbacks of greedy algorithm. • Genetic algorithm performs with better results of 26.14% in comparison with greedy algorithm.
Kabir et al., 2017	Modified Flower Pollination algorithm	• Results are slightly better in terms of fault detection and test suite reduction as compared to other existing concepts considered by authors.
Sugave et al., 2017	Diversity Dragonfly algorithm	• Experimentation is performed with five different programming codes and observed better performance for the DDF approach in comparison with other existing concepts.

B. Regression test Selection

Further, existing work is presented for the regression testing technique of regression test selection. Janhavi and Singh [13] have used UML sequence and state chart diagram and presented the RTSR (Regression Test Selection and Recommendation) model for regression test case selection. Vedpal and Chauhan [14] have applied slicing technique and OPDG concept for the test case selection of object oriented programs. Chauhan et al. [15] have presented the Program Model based Regression Test case Selector (P-ReTEST) approach for the test case selection and evaluated the code based results using open software platform of Graphviz, Cygwin and Eclipse. Do et al. [16] have applied novel test case selection approach on android applications and presented the results with an example of implementation on android application. Refai et al. [17] have used model based approach of FIGA (Fine Grained Adaption) for the test case selection. Dahiya et al. [18] used the UML model based activity, sequence and class diagram for the test case selection. Hafez et al. [19] have used the concept of cache memory to store the information of potential faulty files and test only those selected files. Legunsen et al. [20] have used the static regression selection technique of STARTS (STAtic

Regression Test Selection) which is a Maven plug-in. STARTS approach uses graph based mapping for the selection of test cases by removing faulty cases. Further, Refai et al. [21] have used fuzzy logic based RTS (Regression Test Selection) approach for the selection of test cases. This approach is UML model based approach in which classification is performed on existing test cases to classify reusable and retestable cases as per the changes in the activity diagram. Wongwuttiwat and Lawanna [22] have improved the concept of test case selection by recovering the faulty test cases and reduction of possible test cases. Authors presented the methods of novel proposed model, filtering based selection, code coverage based and original regression test. Romano et al. [23] have used Simple Information Retrieval Regression Test Selection Approach (SPIRITuS) for the test case selection. SPIRITuS outperformed in cost effective and outperformed in comparison with other considered techniques. Table II presents the summarized form of considered existing literature review concepts on Regression Test Selection.

TABLE II EXISTING CONCEPTS OF REGRESSION TEST SELECTION

Author and Year	Used Method	Remarks
Janhavi & Singh, 2014	Regression Test Selection and Recommendation (RTSR)	• Used UML state chart and sequence diagram in the model • Results are evaluated for the case study of ATM system • Reported 61.9% reduction in the test cases in comparison with total number of test cases
Vedpal & Chauhan, 2015	Slicing Technique, Object Oriented Program Dependency Graph (OPDG)	• Focused on all the ways for the regression test case reduction and selection like analysis of affected functions, affected paths and dynamic slicing • Tested the concept for the C++ based programs and observed resulted the reduction in number of test cases.
Chauhan et al., 2015	Program Model based Regression Test case Selector (P-ReTEST)	• Analyzed the results based on small programming codes and observed effectiveness of results. • Reported an improvement of 26.36% in the regression test case selection in comparison with considered Naslavsky's model.
Do et al., 2016	Android application based test case selection	• Presented the novel approach of regression test selection with modules of impact analyzer, coverage generator and test case selector • Applied the approach on android application and presented the results with a working example.
Refai et al., 2016	Fine Grained Adaptation (FiGA)	• Reported the time reduction and test case reduction for test case selection. • Also observed that the fault detection capability of full test set and reduced selected test set are equivalent on the basis of mutation testing.
Dahiya et al., 2016	UML model based sequence, class and activity diagram	• Observed better accuracy for the fault prediction after each semantic change of the regression test selection cases.
Hafez et al., 2016	Potential fault cache test case selection	• Achieved more than 80% results for fault detection and hence the test case selection • Considered approach used static code based approach for bug analysis and dynamic approach to store the link between the classes and test cases
Refai et al., 2017	Fuzzy Logic Based RTS approach (FLiRTS)	• Concept is performed based on UML activity diagram. • Reported comparable results of FLiRTS approach in comparison with code based DejaVu and model based MaRTS approach.
Legunsen et al., 2017	STAtic Regression Test Selection	• Can save time in comparison with ReTestALL approach. Also observed better performance for fault detection
Wongwuttiwat & Lawanna, 2017	Novel proposed model, filtering based selection, code coverage based and original regression test	• Reported the solution of fault avoidance problem along with optimized test cases minimization. • Observed improvement in the size reduction upto 8.55% and fault avoidance upto 2.36% of proposed model in comparison with other considered approaches.
Romano et al., 2018	Simple Information Retrieval Regression Test Selection Approach (SPIRITuS)	• Reported better performance of SPIRITuS in comparison to approaches of Ekstazi, Random-75 and Diff. Approach Outperformed in 24 cases out of considered 42 cases. • Approach is cost effective with a slight compromise with the capability of fault detection.

C. Test Case Prioritization

Test case prioritization technique reorders the execution priority of test cases to improve the quality with maximum assurance. This sub-section presents the existing concepts of test case prioritization. Ansari et al. [24] have used the ant colony based optimization technique for the test case prioritization with the analysis based on size, cost, effort and time of the test cases of prioritized cases and original test cases. Saraswat and Singhal [25] have hybridized the concepts of genetic algorithm and particle swarm optimization for the test case prioritization. Hettiarachchia et al. [26] have used fuzzy expert system based approach for the test case prioritization. Sharma and Singh [27] have performed the test case selection and prioritization using ant colony optimization. Farooq & Nadeem [28] have used while box technique of mutation testing for the test case prioritization with the coverage of additional mutants killed. Ozturk [29] have proposed IMProved Bat Inspired Test Case Prioritization and observed the superiority of results in terms of APFD as compared to other considered techniques. Abid & Nadeem [30] have improved the multiple criteria based approach with "Additional" strategy for the test case prioritization. Chen et al. [31] have used adaptive random sequence based clustering algorithms for the test case prioritization. Table III presents the summarized form of considered existing literature review concepts on Test Case Prioritization.

TABLE III EXISTING CONCEPTS OF TEST CASE PRIORITIZATION

Author and Year	Used Method	Remarks
Ansari et al., 2016	Ant Colony Optimization	• Observed reduction in test case size, effort, time and cost of prioritized test cases in comparison with original test cases. • Achieved better results in terms of Average Percentage of Fault Detection (APFD) of prioritized test cases as compared to non-prioritized test cases.
Saraswat and Singhal et al., 2016	Hybrid approach of Genetic algorithm and Particle Swarm Optimization	• Considered APFD as a fitness function for the genetic algorithm. • Initially genetic algorithm is applied, then detected cases are considered as input for the particle swarm optimization in second phase. • Overall similar results are observed.
Hettiarachchia et al., 2016	Fuzzy Expert System	• Presented a risk based case detection approach • Detect defects at earlier stage with better accuracy as compared to control techniques
Sharma & Singh, 2016	Ant Colony Optimization	• Reported superiority of proposed ACO based approach in terms of APFD.
Farooq & Nadeem, 2017	Mutation Testing	• Higher priority is assigned to the test cases that exposed maximum faults • This higher priority based test considers as the most efficient test case.
Ozturk, 2017	IMProved Bat Inspired Test Case Prioritization (IMPBITCP)	• Reported superiority of IMPBITCP in terms of APFD as compared to BITCP, particle swarm optimization and greedy search approach.
Abid & Nadeem, 2017	Multiple Criteria based test case prioritization	• Multiple criteria approach is modified with "Additional" strategy. • Better performance of proposed multiple criteria based approach as compared to existing multiple criteria and single criteria approach for prioritization of test cases.
Chen at al., 2018	DMClustering (dissimilarity metric clustering with K-medoids), MOClustering_medoids (method object clustering with K-medoids), and MOClustering_means (method object clustering with K-means)	• Reported outperformance of DMClustering in terms of Fm (F-measure), E (distinct detected faults) and APFD • MOClustering_medoids & MOClustering_means achieved better performance results for smaller programs. • Effect of MOClustering_means for smaller programs is more as compared to MOClustering_medoids. • On the other hand, DMClustering performs well for larger programs.

IV. ANALYSIS AND DISCUSSION

In this section, analysis, observations & discussion on the considered research studies is presented with respect to various dimensions of regression testing. The observations have been made on the basis of year wise distribution of publications, and distribution of considered papers for different techniques of regression testing.

A. Distribution of Techniques

In this paper, we have considered the existing work of regression testing techniques of test suite minimization, regression test case selection, and test case prioritization. The distribution of these three techniques is winded up the total 29 reputed research publications. On the basis of considered papers, the distribution of used techniques is shown in fig. 4.

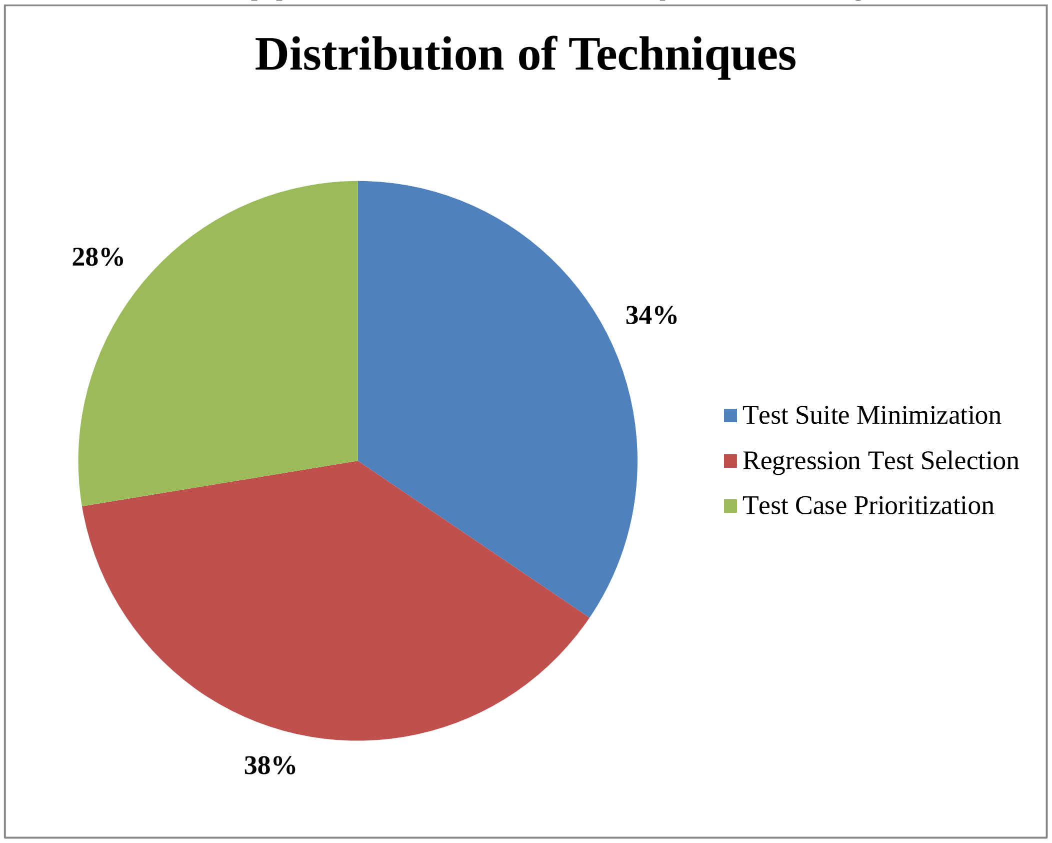


Fig. 4 Distribution of regression testing techniques on the basis of considered papers

B. Distribution of Publications per year

In this paper, we have considered all the three techniques of regression testing with work from 2013 to 2018. Different authors have used different methods for regression testing. This distribution is shown as the publication per year where x-axis defines the publication year and y-axis defines the number of publications. The distribution of considered articles from year 2013 to 2018 is shown in fig. 5.

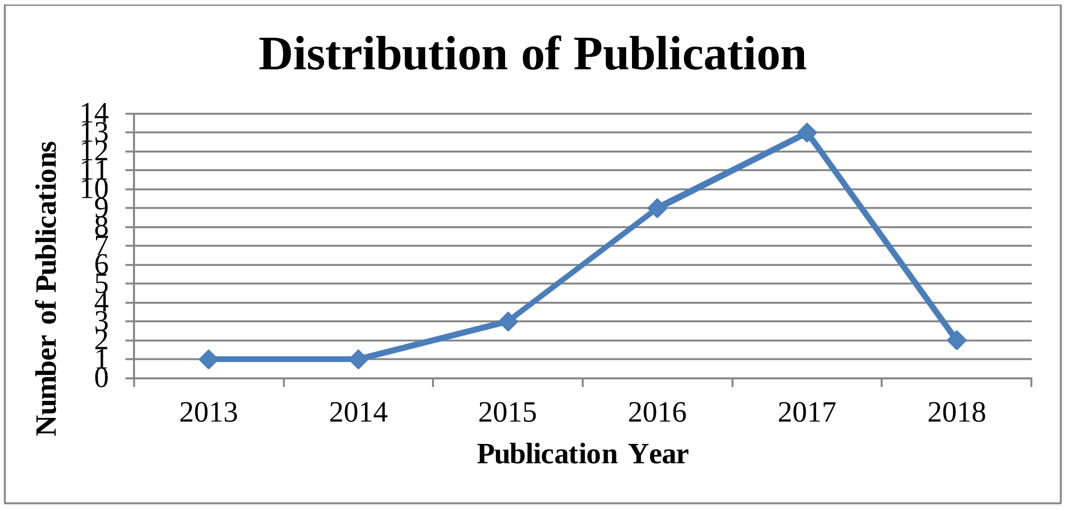


Fig. 5 Distribution of publication on the basis of considered papers

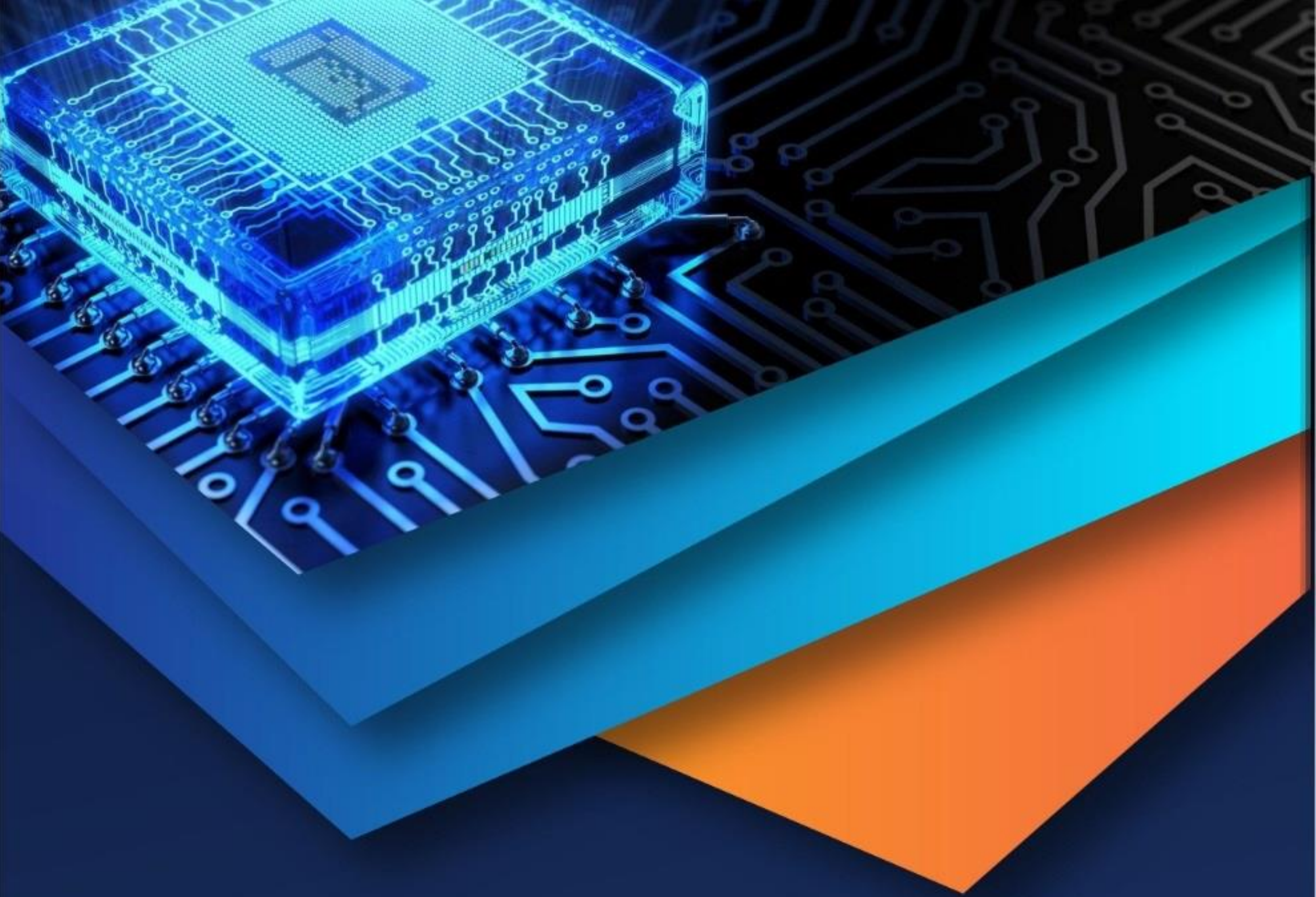
V. CONCLUSION

Advancement in the technology and growing expectation of customers has forced the developer to update or modify the software after a certain period of regular intervals. This addition in the software modules increases the complexity of software. With increasing software complexity, cost and effort required to maintain the software also rises. The risk of bugs and error also increases due to complexity of software modules. This increases the demand of regression testing to ensure the software quality. Even if the software in the existing modules work well but there is always need to retest the software before re-launch in users to maintain the software quality and reputation of the organization belonging to software. In this research papers, the basic of all the regression testing techniques of Retest entire approach, test suite minimization, regression test case selection, and test case prioritization is presented. The existing concepts of these techniques and their analysis and discussion are presented. Based on the analysis and discussion, it can be observed that we have considered all the latest research paper for the review. Overall, different authors have used different methods for different regression testing techniques. It can be observed that researchers have started shifting from traditional approach to computational intelligence and meta-heuristic based approaches due to their advantages of optimization solution and better results.

REFERENCES

[1]. Myers, Glenford J., Corey Sandler, and Tom Badgett. The art of software testing. John Wiley & Sons, 2011.
[2]. Yoo, Shin, and Mark Harman. "Regression testing minimization, selection and prioritization: a survey." Software Testing, Verification and Reliability 22, no. 2 (2012): 67-120.
[3]. Selvakumar, S., T. Manikumar, A. John Sanjeev Kumar, and L. Latha. "An assimilated approach of concept analysis and particle swarm optimization algorithm for effective test suite minimization." In Computational Intelligence and Computing Research (ICCIC), 2013 IEEE International Conference on, pp. 1-4. IEEE, 2013.
[4]. Ahmed, Bestoun S., Taib Sh Abdulsamad, and Moayad Y. Potrus. "Achievement of minimized combinatorial test suite for configuration-aware software functional testing using the cuckoo search algorithm." Information and Software Technology 66 (2015): 13-29.
[5]. Huang, Chin-Yu, Chung-Sheng Chen, and Chia-En Lai. "Evaluation and analysis of incorporating Fuzzy Expert System approach into test suite reduction." Information and Software Technology 79 (2016): 79-105.
[6]. Zhang, Ya-nan, Hong Yang, Zheng-kui Lin, Qing Dai, and Yu-feng Li. "A Test Suite Reduction Method Based on Novel Quantum Ant Colony Algorithm." In Information Science and Control Engineering (ICISCE), 2017 4th International Conference on, pp. 825-829. IEEE, 2017.
[7]. Liu, Feng, Jun Zhang, and Er-Zhou Zhu. "Test-Suite Reduction Based on K-Medoids Clustering Algorithm." In Cyber-Enabled Distributed Computing and Knowledge Discovery (CyberC), 2017 International Conference on, pp. 186-192. IEEE, 2017.
[8]. Indumathi, C. P., and S. Madhumathi. "Cost aware test suite reduction algorithm for regression testing." In Trends in Electronics and Informatics (ICEI), 2017 International Conference on, pp. 869-874. IEEE, 2017.
[9]. Singhal, Shweta, Bharti Suri, and Sanjay Misra. "An empirical study of regression test suite reduction using MHBG_TCS tool." In Computing Networking and Informatics (ICCNI), 2017 International Conference on, pp. 1-5. IEEE, 2017.
[10]. Yamuç, Ali, M. Özgür Cingiz, Göksel Biricik, and Oya Kalıpsız. "Solving test suite reduction problem using greedy and genetic algorithms." In Electronics, Computers and Artificial Intelligence (ECAI), 2017 9th International Conference on, pp. 1-5. IEEE, 2017.
[11]. Kabir, Muhammad Nomani, Jahan Ali, AbdulRahman A. Alsewari, and Kamal Z. Zamli. "An adaptive flower pollination algorithm for software test suite minimization." In Electrical Information and Communication Technology (EICT), 2017 3rd International Conference on, pp. 1-5. IEEE, 2017.
[12]. Sugave, Shounak Rushikesh, Suhas Haribhau Patil, and B. Eswara Reddy. "DDF: Diversity Dragonfly Algorithm for cost-aware test suite minimization approach for software testing." In Intelligent Computing and Control Systems (ICICCS), 2017 International Conference on, pp. 701-707. IEEE, 2017.
[13]. Jahnavi, Singh, Ashima. "Efficient regression test selection and recommendation approach for component based software." In Advances in Computing, Communications and Informatics (ICACCI, 2014 International Conference on, pp. 1547-1553. IEEE, 2014.
[14]. Vedpal, Chauhan, Naresh. "Regression test selection for object oriented systems using OPDG and slicing technique." In Computing for Sustainable Global Development (INDIACom), 2015 2nd International Conference on, pp. 1372-1378. IEEE, 2015.
[15]. Chauhan, Nitesh, Maitreyee Dutta, and Mayank Singh. "A Program Model Based Regression Test Selection Technique for Object-Oriented Programs." In Communication Systems and Network Technologies (CSNT), 2015 Fifth International Conference on, pp. 918-924. IEEE, 2015.
[16]. Do, Quan, Guowei Yang, Meiru Che, Darren Hui, and Jefferson Ridgeway. "Regression test selection for android applications." In Mobile Software Engineering and Systems (MOBILESoft), 2016 IEEE/ACM International Conference on, pp. 27-28. IEEE, 2016.
[17]. Refai Al, Mohammed, Sudipto Ghosh, and Walter Cazzola. "Model-based regression test selection for validating runtime adaptation of software systems." In Software Testing, Verification and Validation (ICST), 2016 IEEE International Conference on, pp. 288-298. IEEE, 2016.
[18]. Dahiya, Sumit, Rajesh K. Bhatia, and Dhavleesh Rattan. "Regression test selection using class, sequence and activity diagrams." IET Software 10, no. 3 (2016): 72-80.
[19]. Hafez, Samia, Mustafa ElNainay, Mohammed Abougabal, and Saleh ElShehaby. "Potential-fault cache-based regression test selection." In Computer Systems and Applications (AICCSA), 2016 IEEE/ACS 13th International Conference of, pp. 1-8. IEEE, 2016.
[20]. Refai Al., Mohammed, Walter Cazzola, and Sudipto Ghosh. "A Fuzzy Logic Based Approach for Model-Based Regression Test Selection." In Model Driven Engineering Languages and Systems (MODELS), 2017 ACM/IEEE 20th International Conference on, pp. 55-62. IEEE, 2017.
[21]. Legunsen, Owolabi, August Shi, and Darko Marinov. "STARTS: STAtic regression test selection." In Proceedings of the 32nd IEEE/ACM International Conference on Automated Software Engineering, pp. 949-954. IEEE Press, 2017.
[22]. Wongwuttiwat, Jittima, and Adtha Lawanna. "Performance improvement model of regression test selection." In Intelligent Informatics and Biomedical Sciences (ICIIBMS), 2017 International Conference on, pp. 49-53. IEEE, 2017.

[23]. Romano, Simone, Giuseppe Scanniello, Giuliano Antoniol, and Alessandro Marchetto. "SPIRITuS: a SimPle Information Retrieval regressIon Test Selection approach." Information and Software Technology (2018).
[24]. Ansari, Ahlam, Anam Khan, Alisha Khan, and Konain Mukadam. "Optimized regression test using test case prioritization." Procedia Computer Science 79 (2016): 152-160.
[25]. Saraswat, Pavi, and Abhishek Singhal. "A hybrid approach for test case prioritization and optimization using meta-heuristics techniques." In Information Processing (IICIP), 2016 1st India International Conference on, pp. 1-6. IEEE, 2016.
[26]. Hettiarachchi, Charitha, Hyunsook Do, and Byoungju Choi. "Risk-based test case prioritization using a fuzzy expert system." Information and Software Technology 69 (2016): 1-15.
[27]. Sharma, Sonia, and Ajmer Singh. "Model-based test case prioritization using ACO: A review." In Parallel, Distributed and Grid Computing (PDGC), 2016 Fourth International Conference on, pp. 177-181. IEEE, 2016.
[28]. Farooq, Faiza, and Aamer Nadeem. "A Fault Based Approach to Test Case Prioritization." In Frontiers of Information Technology (FIT), 2017 International Conference on, pp. 52-57. IEEE, 2017.
[29]. Öztürk, Muhammed Maruf. "Adapting code maintainability to bat-inspired test case prioritization." In INnovations in Intelligent SysTems and Applications (INISTA), 2017 IEEE International Conference on, pp. 67-72. IEEE, 2017.
[30]. Abid, Robeala, and Aamer Nadeem. "A novel approach to multiple criteria based test case prioritization" In Emerging Technologies (ICET), 2017 13th International Conference on, pp. 1-6. IEEE, 2017.
[31]. Chen, Jinfu, Lili Zhu, Tsong Yueh Chen, Dave Towey, Fei-Ching Kuo, Rubing Huang, and Yuchi Guo. "Test case prioritization for object-oriented software: An adaptive random sequence approach based on clustering." Journal of Systems and Software 135 (2018): 107-125.



10.22214/IJRASET

45.98

IMPACT FACTOR: 7.129

IMPACT FACTOR: 7.429

INTERNATIONAL JOURNAL FOR RESEARCH

IN APPLIED SCIENCE & ENGINEERING TECHNOLOGY

Call : 08813907089 (24*7 Support on Whatsapp)