IJSGS FUGUSAU VOL. 7(1) MARCH 2021 WEBSITE: http://journals.fugusau.edu.ng

IJSGS

ISSN: 2488-9229
FEDERAL UNIVERSITY
GUSAU-NIGERIA

INTERNATIONAL JOURNAL OF SCIENCE FOR GLOBAL SUSTAINABILITY

Program Model-Based Regression Tests Selection Approaches: A Review

[1]Samaila Musa and [2]Armiya'u Zango Umar
[1]Department of Computer Science, Federal University Gusau
[2]Department of Mathematical Sciences, Al-Qalam University Katsina

Corresponding Author's Email: samailaagp@gmail.com

Received on: January, 2021 **Revised and Accepted on:** March, 2021 **Published on:** March 2021

ABSTRACT

Regression testing is about running the entire test and is very important activity in software testing. Regression Test Case Selection (RTCS) aims to reduce regression testing cost by only selecting and running the tests that may be affected by code changes. Various RTCS techniques analyzing at either statement, method of class levels has been proposed. This paper presents the result of the study conducted on different Regression Test Case Selection approaches showing mostly stressed areas by researchers and the areas where there is a future scope. Papers related to Regression Test Case Selection techniques in regression testing are reported and documented to find the current trends and future scope for the researchers to work upon.

Keywords: Regression testing, test case selection, software testing, test case, regression test case selection

1.0 INTRODUCTION

1.1 Background of the study

One of the most important activities in software maintenance is regression testing (Zhang *et al*. 2009) that is performed in order to ensure that modifications due to debugging or improvement do not affect the existing functionalities and the initial requirement of the design. According to Rothermel *et al*., (2001) regression testing comprises execution of an enormous amount of tests and is tedious, but it might not be feasible to repetitively re-execute all the test suite in order to tests the software during regression testing due to resource restraints. In this way, it is reasonable to select test case by selecting part of the test suite so that software is tested for early identification of faults. There are numerous regression tests selection techniques, for example, retest-all or randomly selects test cases, however using some pre-defined criteria for selection of test cases might result in more fault identification. There were recently proposed techniques in regression testing (Pandey and Banerjee, 2018; Zhang, 2018; Chen and Zhang, 2018; Jimenez *et al*., 2018; Minhas *et al*., 2018; Tulasiraman and Kalimuthu, 2018; Ngah Munro, and Abdallah, 2017; Dahiya, Bhatia, and Rattan 2016; Sepideh, and Miller, 2015; Panichella *et al*., 2014) in order to improve regression testing performance so as to make it a cost-effective process. These techniques are categorized as test suite minimization, test case se-lection, and test case prioritization (Yoo and Harman, 2012).

Regression test case selection helps in selecting part of tests from the test suite. The most effortless approach is the tester essentially executes all the current tests to guarantee that the new changes are innocuous and is referred as a retest-all technique (Manisha and Singh, 2014). It is the most secure method; however, it is conceivable just if the test suite is small in size. The tests can be chosen arbitrarily to diminish the extent of the test suite. In any case, the greater part of the tests chose arbitrarily can bring about checking little portions of the altered programming, or may not have any connection to the adjusted program. Formally, regression test selection techniques will be an options to the retest-all and random approaches. Regression tests selection techniques for object-oriented programs are grouped into 3 categories by Narula (2016), namely: Firewall-based regression tests selection, Program model-based regression tests selection and Design-based regression tests selection.

2.0 LITERATURE SURVEY

There were a lot of studies performed by researchers and a lot selection techniques were introduced, with the aim of enhance rate of fault detection. These techniques were categorized into several domains according to the nature of the study conducted by the researchers. Manisha and Singh, 2014 conducted a review in which various regression test case selection techniques were examined to enhance th e significance of the selection of test cases from test suite in regression testing. The study conducted by these researchers is elaborated correspondingly into procedural programs. Narula (2016) presented a review on Test Case Selection which is conducted in conferences and journals. But most of the commonly reported techniques are genetic algorithm, adaptive random testing and greedy algorithm.

Code-based RTS methods select tests for testing by performing detail analysis of the source program. It utilizes recovered connections among code portions and tests that navigate them to find test cases for retest when the code is adjusted. Program model-based regression tests selection techniques were proposed in the previous works (Frechette, Badri and Badri, 2013; Larprattanakul and Suwannasart, 2013; Jin, Orso, and Xie, 2010; Tao, Li, Sun and Zhang, 2010; Najumudheen, Mall and Samanta, 2009; El-hamid, El-etriby and Hadhoud, 2010).

El-hamid, El-etriby and Hadhoud, 2010 proposed a technique based on control flow analysis. This technique used Enhanced CFGs (ECFGs) which can be applied to most of the programming languages. But the technique was only able to detect a static change in the software. The technique can be based on generating a set of test inputs that are specifically targeted at the changed code (Jin, Orso, and Xie, 2010), both the two versions of the code are executed with the generated test inputs, can also be on the idea of Control Call Graphs (CCG) (Frechette, Badri and Badri, 2013), reduced form of Control Flow Graph (CFG). The method is more exact and catches the arrangement of calls and allied control than the conventional Call Graph (CC). But the graphs might not be suitable for representation of some OO features.

Other technique (Najumudheen, Mall and Samanta, 2009), constructs graph named call-based object-oriented SDG, instruments the source code with each test case, and various coverage measures are computed from the marked graph. The technique considered in addition to traditional coverage measures, inheritance, and polymorphic coverage. In a similar approach, a technique was proposed by Larprattanakul and Suwannasart (2013) to select test cases for regression testing by using object dependence graph to detect new changes in the source code.

A hierarchical slicing technique for OOP was proposed by Tao *et al*., (2010). The technique computes hierarchy slice on the altered portions of the program, then selects test cases from package level to statements level. This technique sliced from high level to low level of the program.

A regression test selection technique was proposed that is based on analysis of source code of an object-oriented program (Chouhan, Dutta and Singh 2015). A System dependency graph model of the original program was constructed from the source code, and whenever a modification is executed in a program, the constructed model is updated to reflect the changes (Chouhan, Dutta and Singh 2015). The approach in addition to capturing control and data dependencies represents the dependencies arising from object-relations. The empirical studies show that the technique selects on an average of 26.36. % more fault-revealing test cases compared to a Control Dependence Graph based technique while incurring about 37.34% increase in regression test suite size. Dalal and Solanki (2018) proposed an approach that analyze the performance of the BCO-m-GA technique for test case selection against individual non-compounded algorithms GA and BCO. Activity Dependency Graph was used and the paths were served as input for the combination of BCO-m-GA.

Azizi and Do, (2018) proposed a novel language independent Regression test Selection (ReTEST) technique that facilitates a lightweight analysis by

using information retrieval. It uses fault history, test case diversity, and program change history information to select test cases that should be rerun. They evaluated the technique empirically with four open source programs and results show that the approach can be effective and efficient by selecting a far smaller subset of tests compared to the existing techniques.

Regression test selection (RTS) (Zhang, Kim, and Khurshid, 2011; Gligoric, Eloussi, and Marinov, 2015) aims to reduce the cost of regression testing by rerunning only tests whose pass/fail behavior may flip due to the code changes.

In Ngah *et al*. (2019), a new RTCS model called ReTSE was compared with Pythia. The ReTSE model uses decomposition slicing in order to identify the relevant regression tests. While Pythia is a RTCS technique based on textual differencing. Both techniques are compared using a Power program. The analysis of this comparison has shown promising results in reducing the numbers of test cases to be run after changes are introduced. Programs model-based RTCS techniques can be divided based on the elements used such as control-flow based (Binh *et al*., 2017). They developed two main algorithms based on intraprocedural and interprocedural test selection algorithms. The intraprocedural algorithm operates on individual procedures and the interprocedural operates on entire programs or subsystems. In their technique, both the original and modified program were transformed into a CFG in order to perform a comparison.

Panichella *et al*. (2015) propose a RTCS technique based on identifying modified code entities such as functions, variables, types, and macros. Vedpal and Chaulan (2015) have proposed a RTCS technique based on identification of affected paths, affected functions and dynamic slicing which can be used to decrease the number of test cases for regression testing. The technique has two main processes. The first process is the construction of the Object Oriented Program Dependency Graphs (OPDG) for the modified program. The second process is a selection of test cases based on an algorithm that involved dynamic slicing. Qu, Acharya, and Robinson (2012) also proposed a configuration selection technique for regression testing. Their technique addresses the solution to the redundancy issue in retest-all approach for configurable systems and it has employed an existing static slicing tool called *Code Surfer* for computing the configurable option impact and the code change impact.

In Xing *et al*. (2012), a new regression test selection technique was developed based on the program dependence graphs of the original program and its modified version. The approach has entrenched slicing approach in their two main steps. Comparing with previous approaches, they claimed their approach can remove some redundant tests to rerun. All these slicing based RTS approaches are classified as inclusion approaches which select test cases from the test suite that are needed in regression testing. Ngah, Saman, and Munro (2014) developed a new RTCS approach by exclusion using decomposition slicing called ReTSE. Exclusion technique omits redundant test cases from test suite in regression testing

.

Table 1: Program-based Test Case Selection Techniques

YEAR	TEST CASE SELECTION	
	AUTHORS	TECHNIQUES
2009	Najumudheen, Mall and Samanta	Call-based Object-oriented SDG
2010	Jin, Orso, and Xie	Code analysis
2010	Tao, Li, Sun and Zhang	Slicing
2010	El-hamid, El-etriby and Hadhoud	Enhanced Control Flow Graphs (ECFGs)
2011	Zhang, Kim, and Khurshid	Extended Call Graphs (ECG)
2012	Qu, Acharya, and Robinson	Slicing
2012	Xing, Wang, Song and Yang	Program Dependence Graphs (PDG)

2013	Frechette, Badri and Badri	Control Call Graphs (CCG)
2013	Larprattanakul and Suwannasart	Object Dependence Graph (ODG)
2014	Ngah, Saman and Munro	Slicing
2015	Nitesh, Maitreyeea and Mayank	System Dependency Graph (SDG)
2015	Gligoric, Eloussi and Marinov	Dynamic Dependencies
2015	Panichella, Oliveto, Di Penta, and De Lucia	Code analysis
2015	Vedpal, and Chauhan	Object Oriented Program Dependency Graphs (OPDG)
2017	Binh, Duy, Parissis and LusRegTes	Control-Flow Graph (CFG)
2018	Sandeep and Kamna	Use case dependency graph
2018	Azizi, and Do	Fault history
2018	Dalal and Solanki	Activity Dependency Graph
2019	Ngah, Malcolm, Zailani, Masita and, Mohamad	Slicing

The above information is represented in Figure 1. The results show that graphs are mostly used in conducting test case selection as 58%, slicing has the second highest of 21%, while 5% for each fault history, Dynamic Dependency and Code analysis as shown in Figure 1.

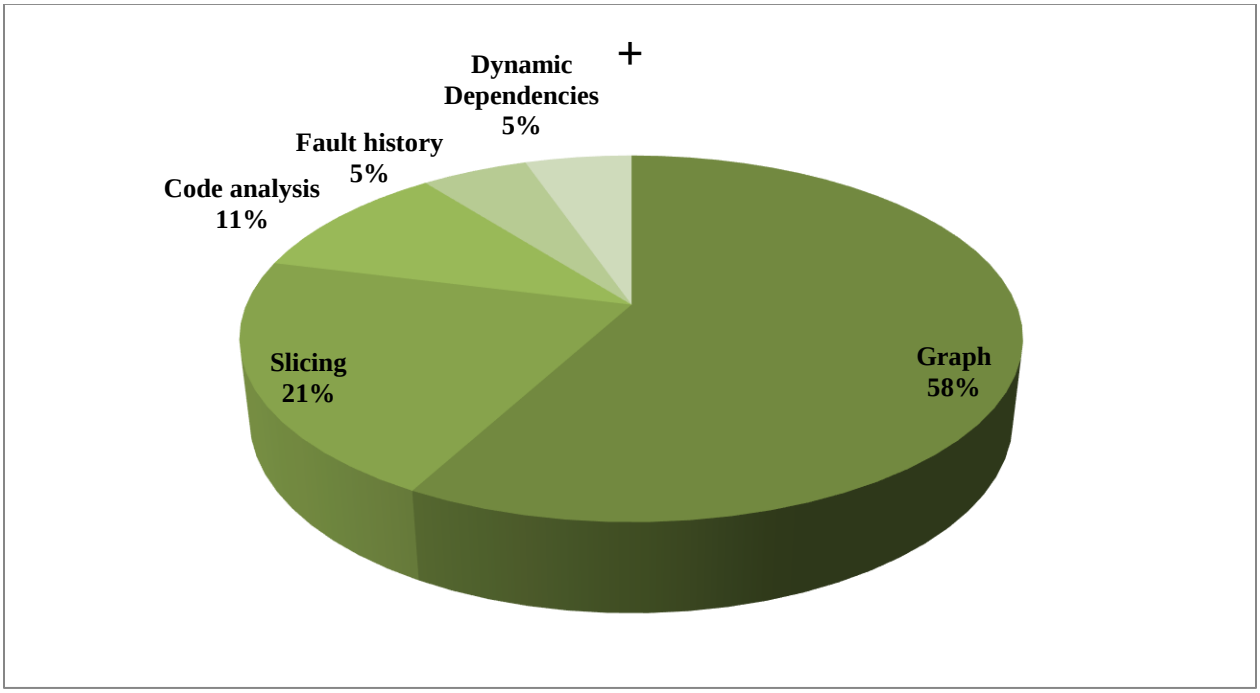

Figure 1: Distribution of various Program-based Test Case Selection Techniques

3.0 CONCLUSION

This paper gives a summary of the program-based regression test case techniques applied to Regression Testing. In future, more research and analysis may be carried out on more techniques which will help to propose new ideas that will give maximum code coverage and minimum execution cost, leading to deliver a product of high quality.

REFERENCES

Ngah, A., Malcolm M., Zailani A., Masita A. J., & Mohamad A. (2019) Regression test selection model: a comparison between ReTSE and pythia. TELKOMNIKA, Vol.17, No.2, April 2019, pp.844~851 ISSN: 1693-6930, accredited First Grade by Kemenristekdikti, Decree No: 21/E/KPT/2018. DOI: 10.12928/TELKOMNIKA.v17i2.10332

Azizi, M., & Do, H. (2018, October). ReTEST: A cost effective test case selection technique for modern software development. In 2018 IEEE 29th International Symposium on Software Reliability Engineering (ISSRE) (pp. 144-154). IEEE.

Binh NT, Duy TC, Parissis. LusRegTes: A Regression Testing Tool for Lustre Programs. International Journal of Electrical and Computer Engineering (IJECE). 2017; 6(5): 2635-2644.

Chen L, & Zhang L. Speeding up mutation testing via regression test selection: An extensive study. Proceedings of the 11th IEEE International Conference on Software Testing, Verification and Validation (ICST 2018). 2018: 58-69.

Dalal, S., & Solanki, K. (2018). Performance Analysis of BCO-m-GA Technique for Test Case Selection. Indian Journal of Science and Technology, 8(1).

El-hamid, W. S. A., El-etriby, S. S. M., & Hadhoud, M. (2010, March). Regression Test Selection Technique for Multi-Programming Language, The 7th International Conference on Informatics and Systems,(pp. 1-5). IEEE

Frechette, N., Badri, L.,& Badri, M. (2013). Regression Test Reduction for Object-Oriented Software: A Control Call Graph Based Technique and Associated Tool. International Scholarly Research Network Software engineering, (pp. 1-10). Hindawi Publishing Corporation.http://dx.doi.org/10.1155/2013/420394

Gligoric, M., Eloussi, L., & Marinov, D. (2015, July). Practical regression test selection with dynamic file dependencies. In Proceedings of the 2015 International Symposium on Software Testing and Analysis (pp. 211-222).

Jimenez I, Watkins N, Sevilla M, Lofstead J, Maltzahn C. Quiho: Automated performance regression testing using inferred resource utilization profiles. Proceedings of the ACM/SPEC International Conference on Performance Engineering. 2018: 273-284.

Jin, W., Orso, A., & Xie T. (2010, April). Automated Behavioural regression testing. Third International Conference on Software Testing, Verification and Validation, (pp. 137–146). IEEE.

Larprattanakul, A.,& Suwannasart, T. (2013, September). An Approach for Regression Test Case Selection using Object Dependency Graph. 5th International Conference on Intelligent Networking and Collaborative Systems,(pp. 617 – 621). IEEE.

Zhang, L Hybrid regression test selection. Proceedings of the IEEE/ACM 40th International Conference on Software Engineering (ICSE 2018). 2018: 199-209.

Manisha R. and Singh, A. (2014). Review of Regression Test Case Selection Techniques, International Journal of Engineering Research & Technology (IJERT), ISSN: 2278-0181.IJERTV3IS051262 www.ijert.org. Vol. 3 Issue 5, May – 2014 CSE Department, DeenBandhuChhotu Ram University of Science and Technology, Murthal, Haryana, India

Minhas NM, Petersen K, Ali N, Wnuk K. Regression testing goals-view of practitioners and researchers. Proceedings of the 24th Asia-Pacific Software Engineering Conference Workshops (APSECW 2017). 2018: 25-31.

Najumudheen, E. S. F., Mall, R., & Samanta, D. (2009, April). A dependence graph-based test coverage analysis technique for object-oriented programs. In Information Technology: New Generations, 2009. ITNG'09. Sixth International Conference on (pp. 763-768). IEEE.

Ngah A, Munro M, Abdallah M. An Overview of Regression Testing. Journal of Telecommunication, Electronic and Computer Engineering (JTEC). 2017; 9(3-5): 45-49.

Ngah A, Saman MY, Munro M. ReTSE: Slicing based Regression Testing. WIT Transactions on Engineering Sciences. 2014; 86: 457-469.

Chouhan, N., Dutta, M., and Singh, M. (2015) A Program Model Based Regression Test Selection Technique for Object-Oriented Programs. September 2015 DOI: 10.1109/CSNT.2015.87

Pandey A, & Banerjee S. Test suite minimization in regression testing using hybrid approach of ACO and GA. International Journal of Applied Metaheuristic Computing. 2018; 9(3): 88-104.

Panichella, A., Oliveto, R., Di Penta, M., & De Lucia, A. (2014). Improving multi-objective test case selection by injecting diversity in genetic algorithms. IEEE Transactions on Software Engineering, 41(4), 358-383.

Qu X, Acharya M, Robinson B. Configuration Selection Using Code Change Impact Analysis for Regression Testing. Proceedings of the 28th

IEEE International Conference on Software Maintenance (ICSM). 2012; 129-138.

Rothermel, G., Roland, H.U., Chu, C., &Harrold, M.J. (2001). Prioritizing test cases for regression testing. IEEE Transactions on Software Engineering, 27(10), 929-948

Narula, S. (2016) Review Paper on Test Case Selection. IJCSET(www.ijcset.net) |April 2016 | Vol 6, Issue 4, 126-128

Sepideh, S. E. & Miller, J. Test Case Prioritization using Extended Digraphs. ACM Transactions on Software Engineering and Methodology (TOSEM). 2015; 25(1): 6:1-6:41.

Dahiya, S., Bhatia, R. K., Rattan, D. Regression Test Selection using Class, Sequence, and Activity Diagrams. IET Software. 2016; 10(3): 72-80.

Tao, C., Li, B., Sun, X., & Zhang, C. (2010, July). An approach to regression test selection based on hierarchical slicing technique. In Computer Software and Applications Conference Workshops (COMPSACW), 2010 IEEE 34th Annual (pp. 347-352). IEEE.

Tulasiraman M, & Kalimuthu V. Cost cognizant history based prioritization of test case for regression testing using immune algorithm. International Journal of Intelligent Engineering and Systems. 2018; 11(1): 221-228.

Vedpal, Chauhan N. Regression Test Selection for Object Oriented Systems using OPDG and Slicing Technique. Proceedings of the 2nd International Conference on Computing for Sustainable Global Development (INDIACom). 2015: 1372-1378.

Xing J, Wang H, Song W, Yang Q. Safe Regression Test Selection Based on Program Dependence Graphs. Proceedings of the 36th IEEE Annual Computer Software and Applications Conference Workshops (COMPSACW). 2012; 188-193.

Yoo, S., & Harman, M. (2012). Regression testing minimization, se-lection and prioritization: a survey. Software Testing, Verification and Reliability, 22(2), 67-120. https://doi.org/10.1002/stv.430.

Zhang, L., Hou, S. S., Guo, C., Xie, T., & Mei, H. (2009, July). Time-aware test-case prioritization using integer linear programming. In Proceedings of the eighteenth international symposium on Software testing and analysis (pp. 213-224). ACM.

Panichella, A., Oliveto, A., Di Penta, M., De Lucia, A., Improving Multi- Objective Test Case Selection by Injecting Diversity in Genetic Algorithms. IEEE Transactions on Software Engineering. 2015; 41(4): 358-383.

Zhang, L., Kim, M., & Khurshid, S. (2011, September). Localizing failure-inducing program edits based on spectrum information. In 2011 27th IEEE International Conference on Software Maintenance (ICSM) (pp. 23-32). IEEE.