

**DYNAMIC SMOKE TESTING
DYNAMIC REGRESSION TEST CASE SELECTION
AND PRIORITIZATION**

Yasitha Nuwan Mallawaarachchi

(168245A)

Degree of Master of Science

Department of Computer Science and Engineering

University of Moratuwa

Sri Lanka

May 2020

**DYNAMIC SMOKE TESTING
DYNAMIC REGRESSION TEST CASE SELECTION
AND PRIORITIZATION**

Yasitha Nuwan Mallawaarachchi

(168245A)

Thesis submitted in partial fulfilment of the requirements for the degree Master of
Science in Computer Science

Department of Computer Science and Engineering

University of Moratuwa

Sri Lanka

May 2020

DECLARATION

I declare that this is my own work and this thesis does not incorporate without acknowledgement any material previously submitted for a Degree or Diploma in any other University or institute of higher learning and to the best of my knowledge and belief it does not contain any material previously published or written by another person except where the acknowledgement is made in the text.

Also, I hereby grant to the University of Moratuwa the non-exclusive right to reproduce and distribute my thesis, in whole or in part in print, electronic or other media. I retain the right to use this content in whole or part in future works.


.....

Yasitha Nuwan Mallawaarachchi

22/05/2020
.....

Date

The above candidate has carried out research for the Masters/MPhil/PhD thesis/
dissertation under my supervision.

.....

Dr. Charith Chitraranjan

.....

Date

ABSTRACT

With the advancement and increasing popularity of agile software development practices in large scale software development projects, frequent product releases are encouraged so that clients can actively participate in the software development life cycle (SDLC) by providing early feedback on developed features. This approach leads to iterative shorter cycles of development and continuous integration. So, the importance of regression testing and regression test suite is well emphasised in such methodologies. Regressions have become the most widely used approach in maintaining the quality of continuously changing software systems.

Even though the agile SDLC requires faster regression feedback given the shorter length of the release cycles, size and the complexity of the regression test suites increases over time; hence execution time keeps on growing. Therefore, it is not practical to run the regression test suite on every code change. In turn, it has become a significant dilemma in current regression testing. Therefore, it is essential to implement a regression testing strategy which is highly selective but accurate, to ensure the committed code changes does not inflict any ill behaviour on the current working software before it is merged and released for client feedback. To achieve this objective, it is critical to find out the distinct effects on behaviour that have impacted the software at the earliest during the continuous integration (CI) cycle. This research is focused on selecting and prioritizing the most suitable test cases from the regression test suite to detect any behaviour that is no longer intact due to the code change. Also, the capability of employing machine learning principles to learn and identify the most impactful characteristics of test cases is considered as another key objective of this study.

Keywords: Regression Test, Selection, Prioritization, Machine Learning, Clustering

ACKNOWLEDGEMENTS

Foremost, I would like to express my sincere gratitude to my advisor Dr Charith Chitraranjan for the continuous support of my MSc study and research, for his patience, motivation, enthusiasm, and immense knowledge. His guidance helped me in all the time of research and writing of this thesis. I could not have imagined having a better advisor and mentor for my MSc study.

I am grateful for the support and advice given by Dr Indika Perera, by encouraging continuing this research till the end. Further, I would like to thank all my colleagues for their help in finding relevant research material, sharing knowledge and experience and for their encouragement.

I am as ever, especially indebted to my parents for their love and support throughout my life. Finally, I wish to express my gratitude to all my colleagues at MillenniumIT Pvt. Ltd, for the support given me to manage my MSc research work.

TABLE OF CONTENTS

DECLARATION	I
ABSTRACT	II
ACKNOWLEDGEMENTS	III
TABLE OF CONTENTS	IV
LIST OF FIGURES	VI
LIST OF TABLES	VII
LIST OF ABBREVIATIONS	VII
1 INTRODUCTION	1
1.1 Agile development practice	1
1.2 Continuous integration and gitflow	2
1.2.1 Continuous integration	2
1.2.2 Git flow	2
1.3 Software maintenance and regression testing	3
1.3.1 Software maintenance	3
1.3.2 Regression testing	5
1.4 Problem addressed by the research	7
1.5 Objectives and expected outcomes of the research	9
1.6 Scope of the research	11
2 LITERATURE REVIEW	13
2.1 Regression test case selection	13
2.1.1 Graph-based regression test case selection	13
2.1.1.1 Graph models	13
2.1.1.2 Regression test case section models for procedural programs	15
2.1.1.3 Regression test case selection model for object-oriented program	17
2.1.2 Code coverage based regression test case selection	20
2.1.3 Specification-based regression test case selection	20
2.2 Regression test case prioritization	22
2.2.1 Test case prioritisation techniques	24
2.2.2 Benefits and challenges of test prioritisation	26

2.3	Usage of machine learning principle for test case selection and prioritization	28
2.3.1	Unsupervised learning based test case minimization	28
2.3.2	Supervised learning based test case minimization	29
2.3.3	Reinforcement learning based test case minimization	29
2.4	Evaluation criteria	30
2.4.1	Evaluation parameters	31
3	METHODOLOGY AND CONCLUSION	34
3.1	Methodology	34
3.1.1	Introduction	34
3.1.2	Machine learning	34
3.1.3	TF-IDF scoring	37
3.1.4	Data clustering	38
3.1.4.1	K-Mean clustering	39
3.1.4.2	Optimal number of clusters	41
3.1.5	System under evaluation	43
3.1.6	Regression test suite	45
3.1.7	Proposed methodology	47
3.1.7.1	Gather test case statistics	48
3.1.7.2	Extract test case features based on the procedure call graph	53
3.1.7.3	Group similar test cases	56
3.1.7.4	Test case selection and prioritization	59
3.2	Evaluation results	63
3.2.1	Evaluation methodology	63
3.2.2	Test case reduction rate and efficiency	65
3.2.3	Precision	66
3.2.4	Recall	68
3.2.5	F – Measure	70
3.2.6	Mutants killed	71
3.2.7	Performance of the test selection levels	72
3.2.8	Summary	72
3.3	Conclusion	74
3.3.1	Outcomes of the research	74

3.3.2	Challenges and limitations	75
3.3.2.1	Challenges	75
3.3.2.2	Research limitations	76
3.3.3	Research assumptions	77
3.3.4	Future work	78
	REFERENCE LIST	80

LIST OF FIGURES

Figure 1.1:	Agile software development life cycle	1
Figure 1.2:	Git flow branching model	3
Figure 1.3:	Software maintenance process	5
Figure 1.4:	Regression test case selection and prioritization	10
Figure 2.1:	Statement level flow graph	13
Figure 2.2:	Firewall technique for D class	18
Figure 2.3:	Specification based regression test case selection	22
Figure 2.4:	Regression test case prioritisation	24
Figure 3.1:	Traditional programming vs Machine learning	34
Figure 3.2:	Supervised learning process	35
Figure 3.3:	Reinforcement learning process	36
Figure 3.4:	Post trade system overview	44
Figure 3.5:	Proposed test case selection and prioritization process	47
Figure 3.6:	TF-IDF score of each test case - 2D graph	55
Figure 3.7:	TF-IDF score of each test case - 3D graph	56
Figure 3.8:	Elbow method output for different K	57
Figure 3.9:	K-mean clustering on top of test cases - 2D graph	58
Figure 3.10:	K-mean clustering on top of test cases - 3D graph	58
Figure 3.11:	Test cases selected for each test group	59
Figure 3.12:	Test case selection precision	67
Figure 3.13:	Test case selection recall rate	70
Figure 3.14:	Evaluation results summary for different test selections	73

LIST OF TABLES

Table 2.1: Test case classification	31
Table 3.1: Post trade system regression test results	46
Table 3.2: Test case execution details extraction process and usage	49
Table 3.3: Test case coverage statistic persisted data	52
Table 3.4: Test case execution statistics persisted data	53
Table 3.5: Test case reduction for different selection methods	65
Table 3.6: Precision comparison based on the individual test cases	67
Table 3.7: Precision comparison based on the test case groups	68
Table 3.8: Recall comparison based on the individual test cases	69
Table 3.9: Recall comparison based on the test case groups	69
Table 3.10: F-measures for different test selection methods	71
Table 3.11: Mutants killed from each test selection methods	71
Table 3.12: Defect detection capability of proposed test selection levels	72
Table 3.13: Evaluation results summary for different test selections	72

LIST OF ABBREVIATIONS

Abbreviation	Description
BBD	Block Branch Diagram
CI	Continuous Integration
CR	Change Request
DB	Database
E2E	End to End
IR	Information Retrieval
ML	Machine Learning
OOP	Object Oriented Programming
ORD	Object Relational Diagram
OSD	Object State Diagram
PCA	Principle Component Analysis

QA	Quality Assurance
SDLC	Software Development Life Cycle
TF-IDF	Term Frequency – Inverse Document Frequency
WSS	Within-cluster Sum of Square