IJCSE International Journal of Computer Sciences and Engineering Open Access
Research Paper Vol.-7, Issue-6, June 2019 E-ISSN: 2347-2693

Test Case Reduction for Object Oriented Systems using Security Metrics

Sameeksha Khare[1*], Rajvir Singh[2], Ajmer Singh[3]

[1,2,3] Department of Computer Science and Engineering, Deenbandhu Chhotu Ram University of Science and Technology, India

Corresponding Author: sameekshakhare65@gmail.com, Tel.: 9991603826

DOI: https://doi.org/10.26438/ijcse/v7i6.371378 | Available online at: www.ijcseonline.org

Accepted: 12/Jun/2019, Published: 30/Jun/2019

Abstract— Test cases play crucial role in software testing. The exhaustive testing of large complex object oriented software systems has been found to be impractical due to large number of test cases execution cost. Due to this reason the researchers in the field of software testing reduce the number of test cases by selecting only effective and important test cases. This paper presents an approach for test case reduction for object oriented systems considering the security as main aspect of software system. Finding out less secure classes help software testers to remove redundant test cases at class level. Object oriented class level security metrics have been considered to detect less secure classes. To evaluate proposed approach, jEdit 5.5.0 software as a case study has been considered. Weka 3.8 was used to generate the proposed mathematical model in order to select effective metrics to detect all the less secure classes. Using feature selection techniques in Weka, effective and ineffective security metrics were categorized. Only effective metrics are taken into account for assigning weighs to each test paths. The results showed the significant improvement in results.

Keywords— *jEdit5.5.0, Test Case, Test Cases Reduction*

I. INTRODUCTION

The definition of Test suite reduction is given as: "Given a test suite TS represents a set of test cases {t1, t2, t3… tn}, a set of test requirements R = {R1, R2 . . . Rn} to be covered, and subsets of TS, S = {S1, S2 . . . Sn}, where each test set is associated with Ri. The objective is to find the representative subset of S that satisfies all of requirements", according to [1].Due to constraints of time and resources, it is essential to make an approach to minimize available test suites. This paper proposes an approach for test case reduction considering the object oriented (OO) security metrics at design level. Due to addition of new test cases to the test suite, size of test suit increases which in turn increases the cost of testing. The overall goal is to reduce the time and cost of testing while maintaining the system secure. To make the system secure, it is very important to find vulnerabilities as early as possible. The less secure classes are identified by finding out the effective OO security metrics. Furthermore, this paper categorizes OO metrics, namely effective and ineffective, linear regression method and feature selection technique. The feature selection technique used is CfsSubsetEval (attribute evaluator) with best first search method. This paper presents a mathematical model used to show the impact of security metrics on OO designs so as to minimize the errors. For the generation of mathematical model, WEKA tool is used.

WEKA stands for "Waikato Environment for knowledge Analysis". It is an open source software which was developed at university of Waikato, New Zealand and is written in Java. It comprises of machine learning approaches for clustering, regression, data classification, rules mining for visualization and association of data (WEKA, 2018).

For analyzing class dependencies of source code of system under test (SUT), Code Pro Analytix version7.1.0 ,a plug-in is used for java eclipse for testers. Various features of this tool include code analysis, JUnit Test Generation, compute metrics, code coverage, dependency analysis and JUnit Test Editor. In this paper, this tool is used for dependency analysis. Once this tool is installed, then for generating class dependency graph (CDG), Right click on the project>CodePro Tools> Dependency Analysis. From this CDG, all the test paths are generated using Breadth First Search (BFS) which is a graph traversal algorithm. BFS visits all the nodes level-wise and generates test paths taking on account source node and destination node. After generating test paths, test cases are selected on the basis of assigned weight to every test path. Then, finally reduced test cases are obtained.

A. Organization of this paper

The organization of paper is as follows. Section I presents the introduction of the field of research, importance of security

in object oriented system and tools used in the research. Section II comprises of existing work on Test Case Reduction in OO systems and few papers on security metrics are also discussed. Section III consists of the proposed methodology used in this research. In Section IV, steps of implementation of used in the approach are presented. Section V comprises of the results and analysis of the research work. Section VI presents conclusion and future work.

II. Related Work

In this section, existing work that is related to Test Case Reduction for Object Oriented systems has been discussed and a brief review of security metrics has been provided.

(DM Thakore and S.J Sarde, 2012) introduced an approach "source code analysis for software quality metrics" to find out code complexity and security of large software and discussed limitations of various tools.

(Soham H Gandhi et al.,2013) developed a new set of OO metrics (CIDA,COA,CMW) to predict vulnerabilities at the design phase taking security metrics based on interaction and accessibility in order to reduce cost of project.

(Nicholas Frechette et al.2013) suggested a control call graph(CCG) technique for regression test case reduction and developed the tool to support the approach used.

(Naresh Chauhan, 2015) presented an approach (OPDP and slicing technique) to reduce number of test cases considering modified object oriented programs.

(Sudhir Kumar Mohapatra and Srinivas Prasad, 2015) presented an ACO Reduce algorithm for test case reduction in Object oriented programs and comparison is done with various other reduction algorithms.

(Bandar M.Alshammari, 2016) introduced a generic model to evaluate the security of object oriented designs by using multilevel classification of its security critical data.

(B.Geetha and D. Jeya Mala, 2016) proposed a novel strategy for Test Case Reduction in Object Oriented Systems using Clustering and Fuzzy Logic.It is a three phase process in which Reflexive technique is used.

(Dr. Sandeep Dalal and Susheela Hooda, 2017) presented combined heuristic approach using Genetic and Fuzzy Clustering Algorithm to test OO Software Systems and to reduce or minimize test cases considering branch coverage criteria.

(Abdullah Al Hussein, 2017) presented OO quality metrics tool for software assessment.

(Allesandro et al., 2017) introduced a multi-objective approach for Test Case Reduction (MORE+) which maximize fault detection. It is implemented for 20 Java applications but the approach was costly.

(Shweta Dwivedi and Santosh Kumar, 2018) proposed an approach of reduction or optimization of test suite for Fuzzy Object Oriented Database (FOOD) taking on account a case study on Hospital Based PDS in which conditions or states of patients is shown using SCD, whole process of Heart Disease is represented using CFG and test cases are reduced using C-Means Clustering Algorithms.

From the above discussion, it is concluded that more work can be done using OO metrics to reduce test cases. In this paper, OO security accessibility metrics are used to remove redundant test cases found using BFS through CDG obtained for software module under test (jEdit5.5.0).

Table1-Test Case Reduction Techniques for OO programs

S.No	Authors	Technique Used	Research gaps
1.	Nicolas Frechette, Linda Badri, and Mourad Badri (2013)	Control call graph based technique	Security metrics are ignored
2.	S.K. Mohapatra and S. Prasad (2015)	ACO reduce technique	Not included security metrics

3.	Naresh Chauhan and Vedpal (2015)	Slicing and OPDG are used	Not discussed security metrics
4.	S.K. Mohapatra (2015)	Genetic Algorithm	Further analysis can be done with program of more line of code in future
5.	B. Geetha And Jeya Mala (2016)	Reflexive Technique	Lack of security metrics Lack of flexibility
6.	B. M. Alshammari (2016)	Model considering cohesion metrics	Influence of software quality properties for example polymorphism, inheritance on the safety of programs that are object-oriented is ignored.
7.	A. Marchetto, G. Scanniello, and A. Susi (2017)	Multi Objective Approach	Focused only on the cost of test suite reduction Absence of the security metrics parameters
8.	Dr Sandeep Dalal, Susheel Hooda (2017)	Used Metaheuristic approach to reduce test cases	Object oriented metrics are not used
9.	A.Al Hussein(2017)	Made Object oriented tool	Quality metrics are considered Ignored the security metrics
10.	S. Dwivedi(2018)	Clustering Technique	Lack of security metrics

III. METHODOLOGY

Relevant details should be given including experimental design and the technique (s) used along with appropriate statistical methods used clearly along with the year of experimentation (field and laboratory).In this proposed work, firstly source code of jEdit 5.5.0 is given as an input then security metrics are calculated for that code.

Secondly, a mathematical model or equation is generated through WEKA tool(free and open source software) using machine learning algorithm namely linear regression which helped in categorizing effective metrics and ineffective metrics. Thirdly, using the same input file, class dependency graph (CDG) was generated using Code Pro Analytics version7.1.0 tool, a plug in used for java eclipse neon.3.

From the CDG, test paths were generated using BFS algorithm. Then weights were assigned to each test path using security prediction mathematical model and sorting of test paths were done based on assigned weights.

Finally, reduced test cases were obtained corresponding to test paths selected. To detect less secure classes, OO security metrics were taken into consideration. Security metrics can be calculated based on factors like interaction, accessibility etc. In this paper, security accessibility metrics has been used.

To examine the values of metrics, jEdit 5.5.0 was chosen. The security metrics considered are Class Public Method Ratio (CPUBMR), Class Private Method Ratio (CPRIMR), Class Protected Method Ratio (CPROMR), Class Public Attribute Ratio(CPUBAR), Class Private Attribute Ratio(CPRIAR), Class Protected Attribute Ratio(CPROAR).

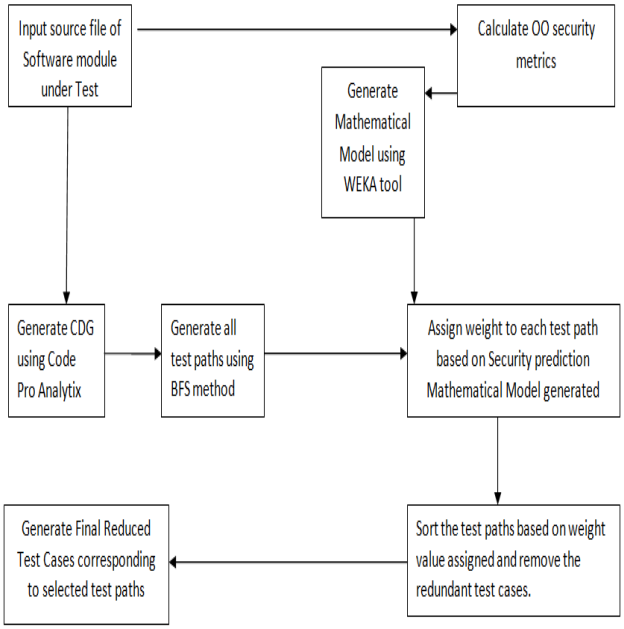


Fig 3.1 Block Diagram for proposed Test Case Reduction Method

The detailed stepwise implementation is given below:

Step1. Selection of effective metrics: Calculate security metrics for software used for evaluation (here jEdit 5.5.0).WEKA provides two methods for selecting effective metrics: (i) Using Filters (ii) Using Select Attributes. CfsSubsetEval (attribute evaluator) with Best First Search Method is used for selecting metrics. The selected metrics are CPUBMR, CPRIAR and Security.

At this step, classifier is selected and mathematical model or equation is generated. Classifier chose for this paper is Linear Regression.

The Linear Regression Model generated is:

*Security Prediction = 2.2293 * CPRIMR + 0.3224 * CPRIAR + 0.4896 * CPROAR - 0.0756*

Root relative squared (R2) error for this model is 88.267%.

Step2. Generation of CDG: Class Dependency graph (CDG) has been used to show the dependencies of classes which will further required to generate test paths through Breadth First Search(BFS). For making class level CDG, Code Pro Analytix version 7.1.0 was used.

Step3. Selection of Test Paths: After generating test paths, selection of test paths take place on the basis of weights calculated for test paths obtained, with the help of model generated through WEKA.

Step4: Generation of reduced test paths: After getting the selected test cases, redundant test cases get removed and 50%test cases having weights of lower value are generated. Finally, reduced test cases are obtained.

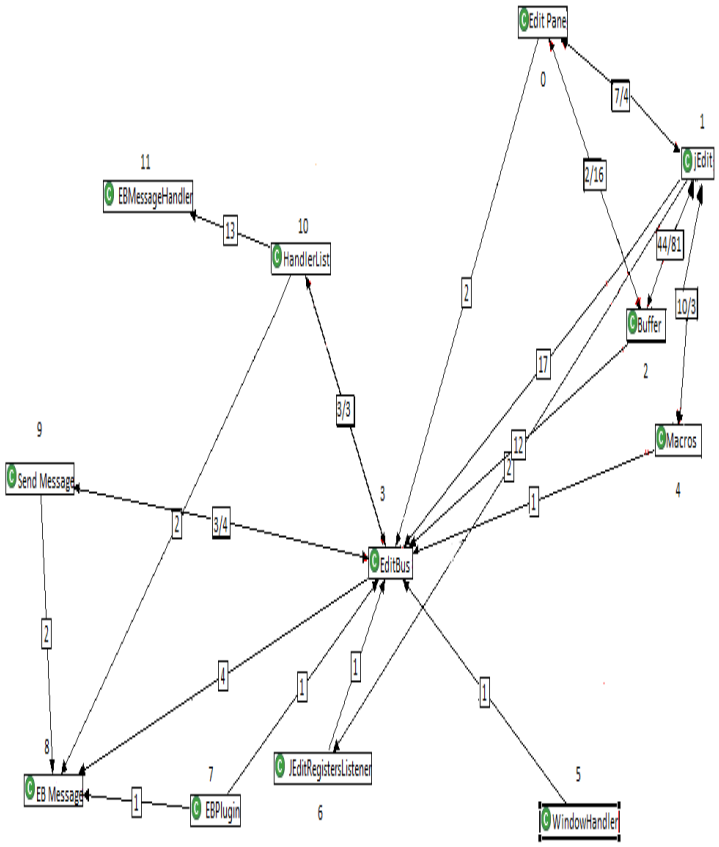
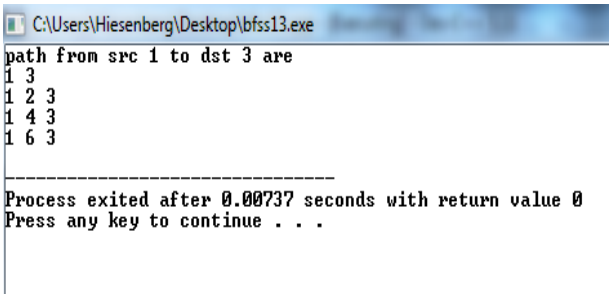


Fig.2. CDG for classes of jEdit5.5.0 module

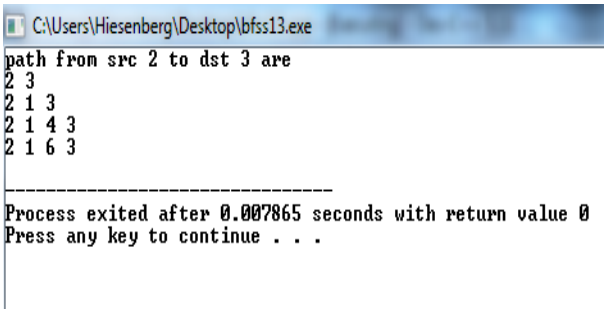
```
C:\Users\Hiesenberg\Desktop\bfss13.exe
path from src 0 to dst 3 are
0 3
0 1 3
0 2 3
0 1 2 3
0 1 4 3
0 1 6 3
0 2 1 3
0 2 1 4 3
0 2 1 6 3

--------------------------------
Process exited after 0.01782 seconds with return value 0
Press any key to continue . . .
```

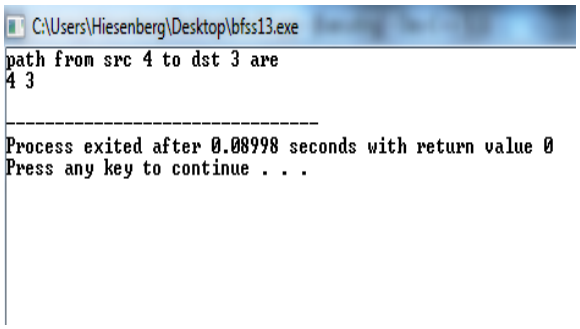
(a)



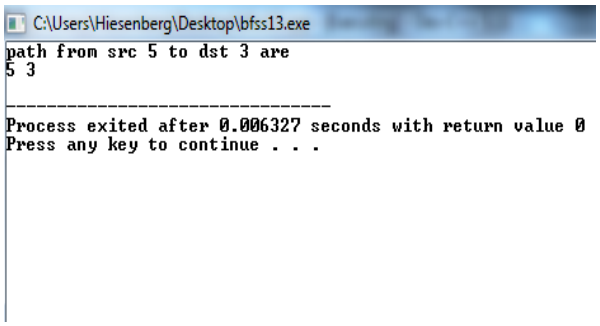
(b)



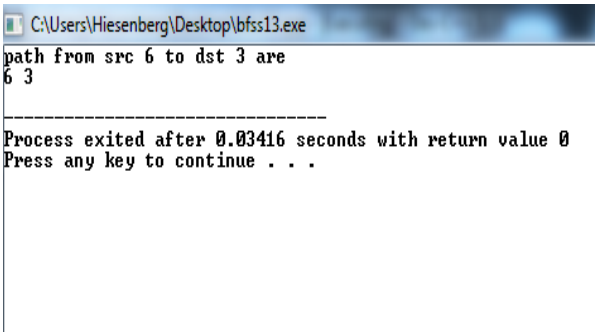
(c)



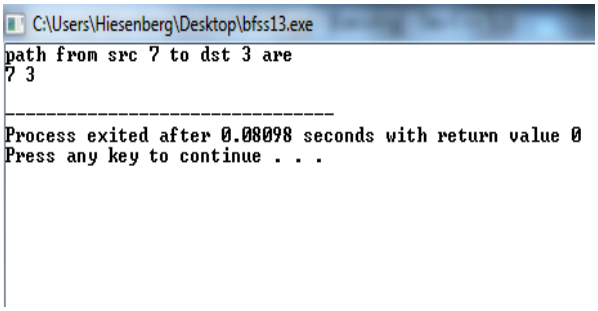
(d)



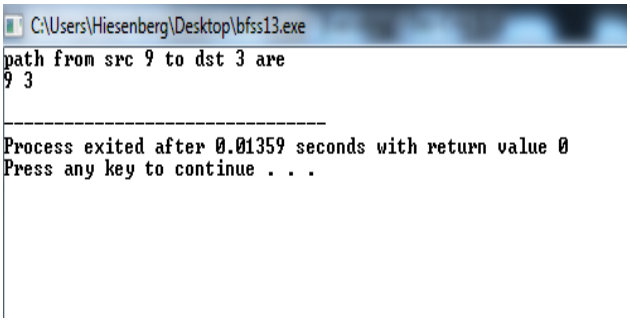
(e)



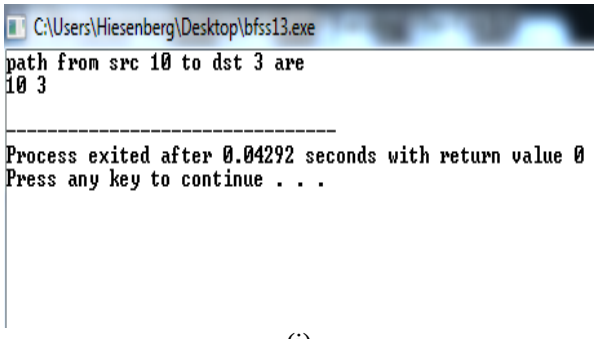
(f)



(g)



(h)



(i)

Fig.3. (a)-(i) Generation of Test paths from CDG using Breadth First Search

These are the test paths that are generated using Breadth First Search from CDG of jEdit5.5.0 software.

Table.2. Test Paths generated using CDG

Test Case ID	Test Paths
TC1	0,3
TC2	0,1,3
TC3	0,2,3
TC4	0,1,2,3
TC5	0,1,4,3
TC6	0,1,6,3
TC7	0,2,1,3
TC8	0,2,1,4,3
TC9	0,2,1,6,3
TC10	1,3
TC11	1,2,3
TC12	1,4,3
TC13	1,6,3
TC14	2,3
TC15	2,1,3
TC16	2,1,4,3
TC17	2,1,6,3
TC18	4,3
TC19	5,3
TC20	6,3
TC21	7,3
TC22	9,3
TC23	10,3

Table 3. Weights Calculated for test cases

Test Case Id	Weight
TC1	0.840185
TC2	1.527165
TC3	1.270024
TC4	1.957004
TC5	1.451565
TC6	1.451565
TC7	1.957004
TC8	1.881404
TC9	1.881404
TC10	1.218440
TC11	1.648279
TC12	1.14284
TC13	1.14284
TC14	0.961299
TC15	1.648279
TC16	1.572679
TC17	1.572679
TC18	0.45586
TC19	1.09396
TC20	0.45586
TC21	0.77826
TC22	1.06292
TC23	1.06292

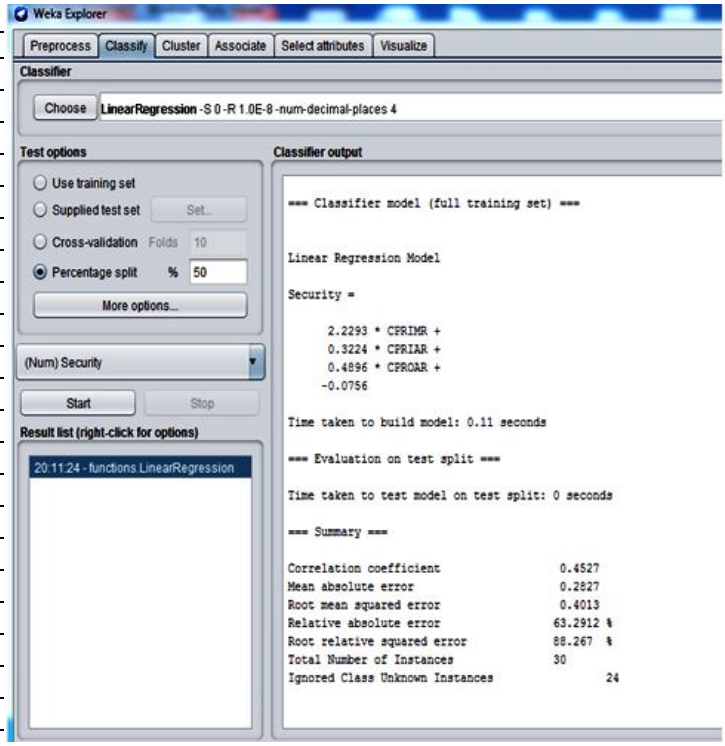


Figure 4: Security prediction Model using WEKA

Table 4. Ordered Test Cases

Test Case ID	Weight
TC18	0.45586
TC20	0.45586
TC21	0.77826
TC1	0.840185
TC14	0.961299
TC22	1.06292
TC23	1.06292
TC19	1.09396
TC12	1.14284
TC13	1.14284
TC10	1.218440
TC3	1.270024
TC5	1.451565
TC6	1.451565
TC2	1.527165
TC16	1.572679
TC17	1.572679
TC11	1.648279
TC15	1.648279
TC8	1.881404
TC9	1.881404
TC4	1.957004
TC7	1.957004

Table 5. Selected Test Paths

Test Case ID	Weight	Nodes
TC18	0.45586	4,3
TC20	0.45586	6,3
TC21	0.77826	7,3
TC1	0.840185	0,3
TC14	0.961299	2,3
TC22	1.06292	9,3
TC23	1.06292	10,3
TC19	1.09396	5,3
TC12	1.14284	1,4,3
TC13	1.14284	1,6,3
TC10	1.218440	1,3
TC3	1.270024	0,2,3

Table 6. Finally Generated Test Cases

Test Case ID	Nodes	Class level test case
TC18	4,3	"org.gjt.sp.jedit.Macros"→ "org.gjt.sp.jedit.Editbus"
TC20	6,3	"org.gjt.sp.jedit.JEditRegisters Listeners"→"org.gjt.sp.jedit.Editbus"
TC21	7,3	"org.gjt.sp.jedit.EBPlugin→ "org.gjt.sp.jedit.Editbus"
TC1	0,3	"org.gjt.sp.jedit.Edit pane"→ "org.gjt.sp.jedit.Editbus"
TC14	2,3	"org.gjt.sp.jedit.buffer→ "org.gjt.sp.jedit.Editbus"
TC22	9,3	"org.gjt.sp.jedit.Send Message"→ "org.gjt.sp.jedit.Editbus"
TC23	10,3	"org.gjt.sp.jedit.HandlerList" → "org.gjt.sp.jedit.Editbus"
TC19	5,3	"org.gjt.sp.jedit.Window Handler"→ "org.gjt.sp.jedit.Editbus"
TC12	1,4,3	"org.gjt.sp.jedit.jEdit"→ "org.gjt.sp.jedit.Macros"→ "org.gjt.sp.jedit.Editbus"
TC13	1,6,3	"org.gjt.sp.jedit.jEdit"→ "org.gjt.sp.jedit.JEditRegisters Listeners"→"org.gjt.sp.jedit.Editbus"
TC10	1,3	"org.gjt.sp.jedit.jEdit" → "org.gjt.sp.jedit.Editbus"
TC3	0,2,3	"org.gjt.sp.jedit.Edit Pane"→ "org.gjt.sp.jedit.buffer" → "org. gjt. sp. jedit.Editbus"

IV. RESULTS AND DISCUSSION

It should include important findings discussed briefly. Wherever necessary, elaborate on the tables and figures without repeating their contents. Interpret the findings in view of the results obtained in this and in past studies on this topic. State the conclusions in a few sentences at the end of the paper. However, valid colored photographs can also be published.

Root Relative Squared Error for proposed model is 88.267%

Root Relative Absolute Error is 63.2912%

Execution time=The execution time is reduced as test cases that are selected are less than the total number of test cases. Assuming execution time for each test case is per unit time(one second) then time saving is 23-12=11s i.e. 47.8% time is saved.

TSI(Total Security Index)=The total security index which was earlier 30.034435,now is reduced to 11.485408 which means the overall security of object oriented system is increased by 38%.

V. CONCLUSION AND FUTURE SCOPE

In this paper, less secure classes are found out for jEdit 5.5.0 software using effective security metrics. CDG is generated for it and all the test paths are revealed using BFS and then weights it and all the test paths are revealed using BFS and then weights are given to each test path using generated model and finally test cases are reduced by removing redundant test cases The approach used for test case reduction is effective as it shows root relative squared (R2) error for corresponding model as 88.267% and 47.8% time is saved and efficient as security is increased by 38%.

The proposed approach used is capable of achieving the objectives but still more work can be done like:

In this case, more options of machine learning can be used for saving more execution time and to automate testing completely.Total Security Index of this approach can be improved by considering more factors of security so that overall security of the object oriented systems can be improved.

REFERENCES

[1] M. Harrold, R. Gupta and M. Soffa, "A methodology for controlling the size of a test suite," ACM Transactions in Software Engineering and Methodology, Vol. 2, No. 3, 1993, pp. 270-285.

[2] A. Agrawal, S. Chandra, and R.A. Khan, *"An Efficient Measurement of Object-Oriented Design Vulnerability",* In Proceedings of International Conference on availability, Reliability and Security, Fukuoka, Japan, 1619 March 2008, ARES 2009

[3] Zhang Guangquan and Rong Mei, *"An Approach Of Concurrent Object-Oriented Program Slicing Based On LTL Property",* International Conference on Computer Science and Software Engineering, pp. 650-653,2008

[4] S. Chandra and R. A. Khan, *"Software Security Metric Identification Framework"* International Conference on Advances in Computing, Communication and Control, pp. 725–731, 2009

[5] B. Alshammari, C. Fidge, and D. Corney (2009), *"Security Metrics for Object-Oriented Class Designs,"* 9th International Conference on Quality Software, Jeju, pp. 11-20,2009.

[6] B. Alshammari, C. Fidge, and D. Corney (2010), *"Security Metrics for Object-Oriented Designs,"* 21st Australian Software Engineering Conference, pp 55-64, 2010.

[7] D. M. Thakore and S. J. Sarde, *"Assessing the Software Complexity and Security metrics from UML Class diagram,"* International Journal of Engineering Research and Applications, vol. 2, no. 4, pp. 585–587, 2012.

[8] S. H. Gandhi, D. R. Anekar, M. A. Shaikh, and A. A. Salunkhe, *"Security Metric for Object Oriented Class Design- Result Analysis,"* International Journal of Innovative Technology and Exploring Engineering, vol no. 6, pp. 139–144, 2013.

[9] N. Frechette, L. Badri, and M. Badri, *"Regression Test Reduction for Object-Oriented Software : A Control Call Graph Based Technique and Associated Tool,"* Hindawi Publishing Corporation ISRN Software Engineering, 10 pages, vol. 2013.

[10] S. H. Gandhi, D. R. Anekar, M. A. Shaikh, and A. A. Salunkhe, *"Finding Accessibility and Interaction Vulnerability of Rational Rose Class Design Using Design Metrics,"* International Journal Of Scientific and Engineering Research, vol. 4, pp. 1–5, 2013.

[11] Devendra Singh Thakore and Dr. Akhilesh R Upadhyay, *"A System for Identification and Assessment of Secure Design using Dynamic Security Metrics,"* Journal Of Information, Knowledge and Research in Computer Engineering, vol. 2, pp. 276–278, 2013

[12] Vedpal, N. Chauhan, *"Regression Test Selection for Object Oriented Systems Using OPDG and Slicing Technique,"* 2nd International Conference on Computing for Sustainable Global Development, pp.1371-1378, 2015

[13] S. K. Mohapatra and S. Prasad, *"Test Case Reduction Using Ant Colony Optimization for Object Oriented Program,"* International Journal Of Electrical and Computer Engineering, vol. 5, no. 6, pp. 1424–1432, 2015.

[14] S. K. Mohapatra and M. Pradhan, *"Finding representative test suit for test case reduction in regression testing,"* International Conference on Computer, Communication and Control, Indore, 2015, pp. 1-6.

[15] S. A. Khan and R. A. Khan, *"Security Improvement of Object Oriented Design using Refactoring Rules,"* International Journal of Modern Education and Computer Science, vol. 2, pp. 24–31, 2015.

[16] B. M. Alshammari, *"A Generic Model for Assessing Multilevel Security-Critical Object-Oriented Programs,"* International Journal Of Advanced Computer Science and Applications, vol. 7, no. 11, pp. 419-427, 2016.

[17] B. Geetha and D. Jeya Mala, (2016) *"Automatic Test Case Reduction in Object Oriented System Using Clustering and Fuzzy Logic,"* Asian Journal of Information Technology, Vol. 15, no. 20, pp. 4071-4076.

[18] A. Al Hussein, *"An Object-Oriented Software Metric Tool to Evaluate the Quality of Open Source Software,"* International Journal Of Computer Science And Network Technology, vol. 17, no. 4, pp. 345–351, 2017.

[19] P. Bhandari, *"Review of Object-Oriented Coupling Based Test Case Selection In Model Based Testing,"* International Conference on Intelligent Computing and Control Systems, pp. 1161–1165, 2017.

[20] A. Marchetto, G. Scanniello, and A. Susi, *"Combining Code and Requirements Coverage with Execution Cost for Test Suite Reduction,"* IEEE Transactions on Software Engineering, vol. 5589, no. c, pp. 1–28, 2017.

[21] S. Dwivedi, *"Minimization of Test Suites for Fuzzy Object-Oriented Database,"* International Journal Of Computer Applications, vol.179, no. 43, pp. 10–15, 2018.

Authors Profile

Mrs. Sameeksha Khare pursed Bachelor of Technology from PIET,Kurukshetra University , Kurukshetra in 2015 and currently pursuing Master of Technology from Deenbandhu Chhotu Ram University of Science and Technology,Murthal. Her main research work focuses on, Test Case Reduction.

Mr Rajvir Singh pursued B.tech and M.Tech in Computer Science and Engineering and is currently pursuing Ph.D. in Software Testing domain and currently working as Assistant Professor in Department of Computer Science, Deenbandhu Chhotu Ram University of Science and Technology , Murthal, India. He is a member of IAENG.He has 10 years of teaching experience and 4 years of Research Experience.

Mr Ajmer Singh pursued B.tech and M.Tech from Kurukshetra University,India. He is currently pursuing Ph.D. in Software Testing domain and currently working as Assistant Professor in Department of Computer Science, Deenbandhu Chhotu Ram University of Science and Technology , Murthal, India. He is a member of IAENG.He has 10 years of teaching experience and 4 years of Research Experience.

