# Coverage DB: A Tool for Intelligent Selection of Tests

Aditya Akotkar
ME Student,
Department of Computer Engineering,
Pune Institute of Computer Technology, Pune, India

M. S. Wakode
Professor
Department of Computer Engineering,
Pune Institute of Computer Technology, Pune, India

## ABSTRACT
Regression testing is an expensive testing procedure utilized to validate modified software. Tester struggles to selectively run the relevant tests for pre-testing defects in software. Standard set of tests identified based on features may not include all the impacted tests for pretesting a fix made at different layers. Also, it is inefficient to execute all the tests for a small change in code. Thus to reduce the cost of testing and to improve the effectiveness, there is a need of identifying, selecting and executing impacted tests based on changes in the code. Existing test selection techniques select non modification revealing and redundant tests. Our system identifies changes made in the code and then selects modification revealing tests using proposed 'Hybrid' technique. 'Hybrid' technique selects optimal and relevant number of tests that would provide maximum test coverage with minimal number of tests. Proposed technique uses a combination of 'By Line' and 'By Function' to increase precision. Redundant tests are further reduced with clustering. The idea is to create a database to map the functional tests and C++ code files by collecting coverage data and then grouping tests based on multiple techniques. Finally, integrating this utility into existing testing process for selecting tests based on changes in the code.

## General Terms
Software Testing, Regression Testing, Code Coverage

## Keywords
Regression Test Selection, Clustering.

## 1. INTRODUCTION
Software systems continuously evolve during development and maintenance. Software is changed for a variety of reasons, such as fixing defects, adding new functionality, or improving its performance. While this evolvement of software systems, it is important to check new functionality as well as old functionality is still working correctly. The modified version of the software should behave as intended, and that modifications should not adversely impact its quality. This is tested by regression testing. General approach of regression testing is to test all the test cases. Thus regression testing is expensive. It takes up to 80 percent of testing budget and 50 percent of maintenance cost. Regression testing may waste resources such as tester's time and computation resources. Till date various prioritization methods are proposed. Prioritization helps in early detection of defect but it does not reduce cost of regression testing. Regression test selection is a well-known problem in software testing. Regression test selection techniques select smaller subset of large regression test suite for testing. 'Retest all' approach is not suitable for large test suites as resource requirement is very high. Also it is inefficient to execute all tests for small change in code. Thus it is necessary to trim the test suite.

## 2. LITERATURE SURVEY
Researchers have studied various methods for improving cost effectiveness regression testing by applying various test selection and prioritization methods. Survey of such methods is carried out by Rothermel [13] and many others [11] [8] [6]. Researchers have given different names to different techniques, but the core logic is similar. These techniques can be broadly classified as follows.

### 2.1 By Line
This is very basic test case selection method based on code coverage [14]. In this approach test cases which traverses through modified source lines, are selected. In practical, considering only modified lines may not give impacted test as change in one line may affect others. To solve this, lines are normalized by considering code block of five lines (above two lines and below two lines) and test cases which pass through this block are selected. Tests selected by this approach are large in number.

### 2.2 Max Min
This is a statistical approach to solve regression test selection problem. In this approach minimum number of tests are selected such that their combined coverage is maximum. This technique ensures that most of the code is tested. Tests selected by this method are less than that of By Line. Similar technique is used in various prioritization algorithms. Survey of such techniques is given by Rothermel [5].

### 2.3 Intersect
This is hybrid approach of solving regression test selection problem. This method is combination of By Line and Max Min. Tests which traverse through modified part and provides high coverage are selected. Kandil [1] proposed similar approach which is combination of test case selection and prioritization. As tests are selected by combining above two methods, tests selected are less in number.

### 2.4 Clustering
In addition to above techniques clustering technique can be used in test case prioritization [3] [4]. In clustering technique, test cases are clustered based on certain properties such as code coverage, previous fault history etc. Clustering group similar test cases in one group and dissimilar test cases in different groups. Different techniques ensure that dissimilar tests are selected so as to provide high fault detection capability.

'By Line' technique works on approximation, which is not very reliable in case of coding. By selecting function level granularity, test cases traversing through function which contains modified code can be obtained. By using function level granularity, more number of tests than are selected. Combination of 'By Line' and 'By Function' will select modification revealing test cases. Combined approach will

select large number of redundant tests which can be removed by clustering them on the basis code coverage.

## 3. SYSTEM ARCHITECTURE

Figure 1 describes 'Coverage DB'. System provides a utility to tester for selecting regression test suite. System can be divided into two parts. First is building coverage database and second is integrating it with existing defect system.

### 3.1 Build Coverage DB

System executes test suite on application under test and generates coverage data using open source code coverage tool. 'Open Cpp Coverage' is used as code coverage tool. 'Open Cpp Coverage' is a free and open source code coverage tool for Cpp applications. Code coverage data is generated in xml format. Code coverage data is then loaded into the database. Also data is merged to produce a single report in HTML format. This report can be used by functional testers to add more tests to test suite.

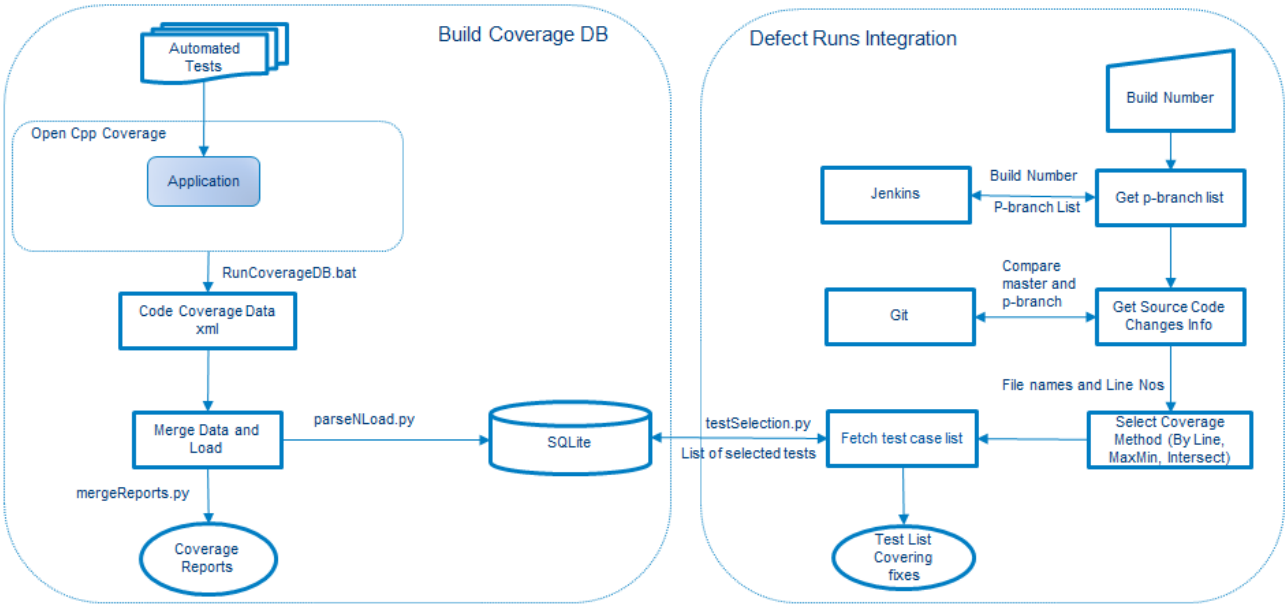


**Fig 1: System Architecture**

### 3.2 Integration with Defect Runs

Second part is integration with defect system. Jenkins build number is taken as input from tester along with the method of test selection. There are four methods provided namely 'By Line', 'Max Min', 'Intersect' and proposed method which is named as 'Hybrid'. To get modified file and line information, developer's branch is compared with master branch using version control system. Diffs obtained are then parsed to extract modified file name and line number. Finally tests are selected according to the selected method.

### 3.3 Proposed Test Selection Method

Figure 2 describes proposed test case selection technique. It combines 'By Line' and 'By Function' method to select modification revealing tests. For modified line which is at function edge, 'By Line' approach may select lines outside the function and thus ineffective tests. Combination of techniques will discard such ineffective tests.

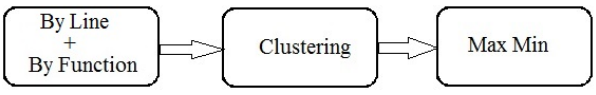


**Fig 2: Proposed Test Selection Method**

**Algorithm:**

1. Execute test suite on the code under test.
2. Generate code coverage report at line level and function level granularity.
3. Select regression test suite based on 'By Line' method.
4. Trim test suite obtained in step 3 by selecting matched test with 'By Function' method.
5. Apply clustering technique on trimmed test suite to remove redundant tests.
6. Apply 'Max Min' on each cluster to get final reduced test suite.

Clustering is done on the basis of code coverage. Execution trace of each test case is represented by a binary string. Each bit corresponds to a statement in the source code. If the statement is executed by the test case, the digit is 1; otherwise it is 0. The similarity between two test cases is measured by the distance between two binary strings using Hamming distance. Output of clustering is groups of similar test cases. These similar test cases can be a same test case with different arguments for checking boundary conditions or different test cases checking same code. Such redundant test cases are removed by clustering. Last step selects test case from each cluster which has maximum coverage.

## 4. RESULTS

Graph shows comparison of proposed test selection method with existing methods. Inclusiveness and precision are comparison parameters given by Rothermel [13]. Selected test percentage is percentage of test suite selected for testing. Redundant test percentage is determined by expert by manually analyzing test suite.

To calculate performance evaluation parameters, test selection methods are applied on multiple known defects. Average values of comparison parameters are plotted on the graph. Graph shows that 100% precision is achieved in proposed test selection technique and up to 10% less redundant tests are selected. Inclusiveness of Hybrid technique is less as it omits redundant tests.

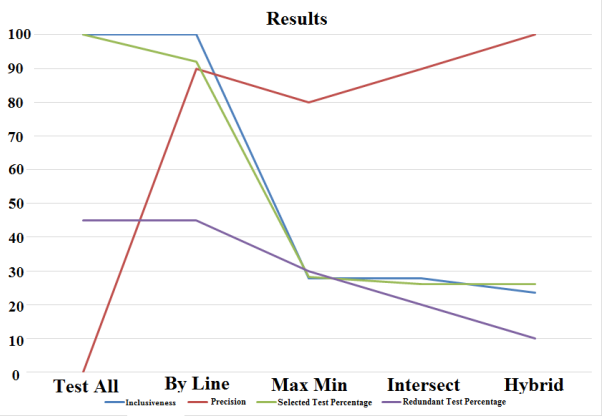


**Fig 3: Results**

## 5. CONCLUSION

Proposed system is useful for testers to select regression test cases from large test suite. It is also useful for developers for quick testing changes in their code. This reduces testing time and results in fast deployment of project. Existing test selection techniques have less precision and more redundancy in selected tests. Proposed Hybrid test selection technique clusters test cases on the basis of code coverage to reduce redundancy. Combining 'By Line' and 'By Function' restricts selection of tests that are not modification revealing which results in increase in precision. This technique is combination of test case selection and prioritization offering advantages of both. 'Hybrid' test selection technique increases precision by 10 percent and selects up to 10 percent less redundant tests than 'Intersect' technique. The research work provides solution for regression test selection problem which has high precision and less redundancy than existing solutions.

## 6. ACKNOWLEDGMENTS

This work is sponsored by SAS Research and Development, Pune. We take this opportunity to express my deep sense of gratitude towards Mr. Kishore Jain, Manager, SAS Research and Development Pune, for giving me this splendid project for my dissertation work. We would like to thank Mr. Atul Kachare, Tech Lead, SAS Research and Development Pune, for his valuable guidance. We wish to express my thanks to Dr. Rajesh Ingle, HOD Computer Department, PICT, for encouragement and providing best facilities. We thank all the staff members for their indispensable support and for most valuable time lent as and when required. We thank all the people who are directly or indirectly involved in this project.

## 7. REFERENCES

[1] Kandil, Passant, Sherin Moussa, and Nagwa Badr. "Regression testing approach for large-scale systems." Software Reliability Engineering Workshops (ISSREW), 2014 IEEE International Symposium on. IEEE, 2014.

[2] Shahid, Muhammad, Suhaimi Ibrahim, and Mohd Nazri Mahrin. "Code Coverage Information to Support Regression Testing." The International Conference on Informatics and Applications (ICIA2012). The Society of Digital Information and Wireless Communication, 2012.

[3] Carlson, Ryan, Hyunsook Do, and Anne Denton. "A clustering approach to improving test case prioritization: An industrial case study." Software Maintenance (ICSM), 2011 27th IEEE International Conference on. IEEE, 2011.

[4] Yoo, Shin, et al. "Clustering test cases to achieve effective and scalable prioritisation incorporating expert knowledge." Proceedings of the eighteenth international symposium on Software testing and analysis. ACM, 2009.

[5] Elbaum, Sebastian, Alexey G. Malishevsky, and Gregg Rothermel. "Test case prioritization: A family of empirical studies." IEEE transactions on software engineering 28.2 (2002): 159-182.

[6] Bharati, Chandana, and Shradha Verma. "Analysis of Different Regression Testing Approaches." Analysis 2.5 (2013).

[7] Kapfhammer, Gregory M. "Empirically evaluating regression testing techniques: Challenges, solutions, and a potential way forward." Software Testing, Verification and Validation Workshops (ICSTW), 2011 IEEE Fourth International Conference on. IEEE, 2011.

[8] Yoo, Shin, and Mark Harman. "Regression testing minimization, selection and prioritization: a survey." Software Testing, Verification and Reliability 22.2 (2012): 67-120.

[9] Blondeau, Vincent, et al. "Test case selection in industry: an analysis of issues related to static approaches." Software Quality Journal (2016): 1-35.

[10] Pathania, Yamini, and Gurpreet Kaur. "Role of Test Case Prioritization based on Regression Testing using Clustering." International Journal of Computer Applications 116.19 (2015).

[11] Biswas, Swarnendu, et al. "Regression test selection techniques: A survey." Informatica 35.3 (2011).

[12] Chittimalli, Pavan Kumar, and Mary Jean Harrold. "Recomputing coverage information to assist regression testing." IEEE Transactions on Software Engineering 35.4 (2009): 452-469.

[13] Rothermel, Gregg, and Mary Jean Harrold. "Analyzing regression test selection techniques." IEEE Transactions on software engineering 22.8 (1996): 529-551.

[14] Beena, R., and S. Sarala. "Code coverage based test case selection and prioritization." arXiv preprint arXiv:1312.2083 (2013).

[15] Huang, Sheng, Jun Zhu, and Yuan Ni. "ORTS: a tool for optimized regression testing selection." Proceedings of the 24th ACM SIGPLAN conference companion on Object oriented programming systems languages and applications. ACM, 2009.