International Journal on Recent and Innovation Trends in Computing and Communication ISSN: 2321-8169
Volume: 5 Issue: 4 109 – 114

# A New Approach For Regression Test Case Prioritization Using Branch Coverage, Decision Coverage And Criticality Coverage Techniques

Sulaxna Solanki
M.Tech. Scholar, Department of Computer Engineering,
Govt. Mahila Engineering College
Ajmer, Rajasthan, India
*sulaxnasolanki19@gmail.com*

Amritpal Singh Yadav
Assistant Professor, Department of Computer Science and Engineering, Govt. Mahila Engineering College
Ajmer, Rajasthan, India
*palamrit83@gmail.com*

***Abstract*—**Regression Testing is the process of retesting the modified parts of the software and checking that no new faults have been created into already existing code. When new features are added to an existing software system, then regression testing is necessary to test the new features as well as the existing features to ensure that their behaviors are not affected by the modifications. Test cases are used to determine whether an application or software system is working correctly or not. It is difficult to re-execute every test case for a program if changes occur. Testers will prioritize the test cases to reduce the cost of regression testing. The main purpose of test case prioritization is to increase the rate of fault detection. In this paper a new hybrid approach for regression test case prioritization is proposed. Excremental results ensure that the new proposed algorithm for test case prioritization improves the rate of fault detection.

***Keywords*-***Software testing, Regression Testing, Test case Prioritization*

*****

## I. INTRODUCTION

Regression testing is a type of software testing that intends to ensure that changes to the software have not affected it. It may include in the functional testing. Regression testing may define as "it is the testing process that is done to find the regression in the system after doing any changes in the product or it is used to check whether the new changes occurs the errors in the software or not". For example, before applying any change on program it must be tested, then again retested the program in the selected areas after a change is applied, to detect whether the change created new bugs or issues, or it may achieve its intended purpose after the actual change is made. Thus, regression testing is essential for large organization. It can be performed during any level of testing. Research in regression testing is a wide range of topics. A test case is a single step, or occasionally a sequence of steps, to test the correct functionalities, features of an application. Earlier work in this area investigated different environments that can assist regression testing. Whenever the software is modified/changed, then regression testing is performed to check whether the new modification on the software can introduce any new errors or not, on the existing software. It is Impractical to re-execute every test case for program if change occurs. It takes lot of time, cost and resources to re-execute all test cases. There are various techniques are commonly used to reduce complexity of regression testing. Automation testing mainly used in regression testing and apart from regression testing, it is also used to test the application from load performance and stress point of view. Automation testing increases the test coverage; improve accuracy, saves time as well as money in comparison to manual testing. Retest all, Test case minimization, test case selection, test case prioritization are the important techniques to improve the effectiveness of regression testing.

**A. Retest All**- In this method, the test cases that no longer apply to modified version of program are discarded and all the remaining set of test cases are used to test the modified program.

**B. Test Case Minimization**- It is a process that seeks to identify and then eliminate the redundant test cases from the test suite.

**C. Test Case Selection** - Test case selection deals with the problem of selecting a subset of test cases that will used to test the changed parts of the software.

**D. Test Case prioritization** - Test case prioritization concerns about perfect ordering of test cases. The prioritization of test cases depending on business impact, and frequently used functionalities. In this method, selection of test cases based on priority will greatly reduce the regression test suite. There are number of matrix available to calculate the fault detection rate using test case prioritization technique. The main goal of the test case prioritization schedule test cases in order to increase their ability to meet some performance goal: Rate of fault detection, Rate of code coverage and rate of increase of confidence in reliability. The rate of fault detection which is a measure of how quickly the fault is detected so that during testing faster feedback can provide about system under testing and allow the software tester to correct the software at earlier phase as possible.

## II. LITERATURE SURVEY

Kaur A. & Bhatt (2011) [1] presents a combined analytic view of evolutionary computation techniques namely Genetic Algorithm and Particle Swarm Optimization. The PSO is an optimization technique, where global solution is constructed by analysis of the local optimal solution. Li K.& Zhang Z. et al proposes (2010) [2] a new method to breeding software test data called GPSMA for structure data test generation, introducing a new strategy to replace the mutation operation in traditional genetic algorithm, and using the "excellent rate of production" to implement the interaction between sub-populations. Kire K. & Malhotra [3] (2014) has worked on Optimization algorithms based on swarm intelligence can have some distinct advantages over traditional methods. It has been found that most SI-based algorithms use mutation and selection to achieve exploration and exploitation. Rini D. &Siti et al (2011) [4] have made review of the different methods of PSO algorithm. The process of PSO algorithm in finding optimal values follows the work of an animal society which has no leader. Particle swarm consists of a swarm of particles, where the particle represents a potential solution. Particle will move through a multidimensional search space to find the best position. Sharma C. &Sabharwal (2013) [5] has worked on applications of GA in different types of software testing are discussed. The GA is also used with fuzzy as well as in the neural networks in different types of testing. It is found that by using GA, the results and the performance of testing can be improved. Tandon A. &malik (2012) [8] has worked on the preliminary results from a genetic algorithm based approach to software test case breeding. The guiding fitness function can provide a focused search that produces a large number of localized test cases, or be loosened up for more random-like behaviors, depending on the testing scenario. High volume or long sequence testing involves the repeated execution of a substantial number of test cases. Windisch A. & Joachim et al (2007) [9] has reported on the application of particle swarm optimization to structural software testing. Both particle swarm optimization and genetic algorithms were used to automatically generate test cases for the same set of test objects. Yang X. (2014) [10] has worked on this ABC technique is used for the generation of the test data. The parallel behavior of the bees makes generation of test cases faster and efficient. Here, independent test path coverage criterion is used as objective criteria to achieve the all test coverage with less number of test runs.

## III. PROPOSED WORK

In this work we have taken following criteria's for test case prioritization.

Statement Coverage
Branch Coverage
Criticality Coverage
The details are as follows:

Statement Coverage Weight SCW- The number of statements a test case cover is calculated. Statement Coverage Weight is equal to the number of statements of the source code are covered by a test case. To calculate the Statement Coverage Weight, annotations are used in the source code written in java programming language.

Branch Coverage Weight BCW - The number of branches or conditions a test case cover is calculated. Branch Coverage Weight is equal to the number of branches of the source code are covered by a test case. To calculate the Branch Coverage Weight, annotations are used in the source code written in java programming language.

Criticality Coverage Weight CCW- The number of critical features (The features which are most important for user point of view. These features are taken as input from the user at the time of requirement analysis and should be mentioned in Software Requirement Specification SRS) Criticality Coverage Weight is equal to the number of critical features of the SRS are covered in source code. To calculate the Criticality Coverage Weight, annotations are used in the source code written in java programming language.

Prioritized Weight PW - To find the order of a test case in prioritized test suite, a prioritized weight is calculated. This prioritized weight is the average of all the three weights i.e. Statement Coverage Weight SCW, Branch Coverage Weight BCW and Criticality Coverage Weight CCW.

In this paper a new algorithm to prioritize regression test suite is proposed. The algorithm calculate three types of weight i.e. statement coverage weight, Branch Coverage Weight and Criticality Coverage Weight and then combine these three types of weights to find Prioritized Weight. All the test cases in the test suite are sorted in descending order of the prioritized weight to find the prioritize regression test suite. The proposed algorithm to prioritize regression test suite is given in Algorithm-1. The algorithm take un-prioritized regression test suite as input and return a prioritized regression test suite.

**Algorithm-1 : Hybrid Algorithm using Statement Coverage, Branch Coverage and Criticality Coverage**

Input : Un-prioritized Regression Test Suite (URTS)
Output : Prioritized Regression Test Suite (PRTS)

Step-1 :Execute all test cases in Un-prioritized Regression Test Suite (URTS) and calculate the value of Statement Coverage Weight SCW, Branch Coverage Weight BCW and Criticality Coverage Weight CCW.

Step-2:Add the value of SCW, BCW and CCW and find the average of the three weight to get Prioritized Weight PW.

Step-3:Sort all the test case in decreasing value of Prioritized Weight PW to get Prioritized regression test Suite (PRTS).

The Algorithm-1 has been implemented in JAVA for a given un-prioritized regression test suite and the effectiveness of the algorithm is checked using well known metric Average

percentage of fault detection APFD. APFD metrics is used to calculate the level of effectiveness which is as follows.

Average percentage of fault detection (APFD) metric-

The goal of increasing a subset of test suite's rate of fault detection, we use a metric called APFD that measures the rate of fault detection per percentage of test suite execution. The APFD is calculated by taking the weighted average of the number of fault detected during the run of the test suite. APFD can be calculated as follows:

APFD = 1- ((tf1+tf2+……+tfm)/n*m +1/2*n)

n is the number of test cases

m is the number of faults

(tf1+tf2+……..+tfm) are the position of first test T that exposes the fault.

This formula represents that we can calculate APFD only when we have prior knowledge of faults contained in the program.

## IV. RESULT ANALYSIS

In this work an un-prioritized regression test suite is generated (for a given sample problem and program). The test suite is having 20 test cases. The value of APFD metric for Un-prioritized Regression Test Suite (URTS) is 70.83 After prioritizing the test suite using Algorithm-1 the value of APFD metric for Prioritized Regression Test Suite (PRTS) is calculated and it is found 94.16 which is 23.33% better than the URTS. Figure-1 is showing a snapshot of the calculation of APFD for URTS which comes out 70.83. Figure-2 is showing a snapshot of APFD for PRTS which comes out 94.16. Figure-3 shows a bar chart for comparing the value of APFD for URTS and PRTS. It shows that APFD for PRTS is better than URTS.

## CONCLUSION AND FUTURE SCOPE

This paper describes a new approach for regression test case prioritization using branch coverage, decision coverage and criticality coverage techniques. It uses statement coverage, branch coverage and criticality coverage criteria's to prioritize regression test suite. Statement and branch coverage ensures that the test cases which covers maximum part of the source code must be tested first. Criticality coverage ensures that the test case which are testing most critical component/functionality of the source code must be executed first. The knowledge about the critical components of the software is collected at the time of requirement analysis and should mentioned in the SRS. The Algorithm is implemented and tested using JAVA on a test suite of 20 test cases. In future the performance of the proposed algorithm can be tested on a larger test suite having thousands of test cases. Also the Algorithm can be tested on a source code having thousands of line of code. In future some other criteria's such as sample of successful test cases sample of failure test cases can also be combined and the performance can be tested.

## ACKNOWLEDGMENT

The This is opportunity to express my heartfelt words for the people who were part of this thesis in numerous ways, people who gave me unending support right from beginning of the thesis.First and foremost I extend my thanks and gratitude to my thesis guide, Mr. Amritpal Singh Yadav (Asst. Professor Department of Computer Science and Engineering) Govt. Mahila Engineering College, Ajmer whose guidance, teaching and certain suggestion provided me the timely valuable input which enhanced my knowledge and thus helped in the development of this Dissertation part-II.I would like to give sincere thanks to the Principal, Dr. Ajay Singh Jethoo for his valuable support. I also extend my thanks to Mrs. PayalAwwal Head of the Department (Computer Science Engineering) for the cooperation and guidance.

## REFERENCES

[1] Kaur Dr. Arvinder, Divya Bhatt"Hybrid Particle Swarm Optimization for Regression Testing" IJCSE, Vol. 3 No. 5(2011).

[2] Li Kewen, Zilu Zhang, Jisong Kou"Breeding Software Test Data with Genetic- Particle Swarm Mixed Algorithm "Journal Of Computers, VOL. 5, NO. 2(2010).

[3] Kire Kevilienuo, Neha Malhotra "Software Testing using Intelligent Technique "IJCA, (2014).

[4] Rini Dian palupi, siti mariyam shamsuddin et al"Particle Swarm Optimization: Technique, System and Challenges" IJCA, Volume 14, (2011)

[5] Sharma Chayanika, Sangeeta Sabharwal et al "A Survey on Software Testing Techniques using Genetic Algorithm "IJCSI, (2013).

[6] Singh Abhishek, Naveen Garg et al" A Hybrid Approach of Genetic Algorithm and Particle Swarm Technique to Software Test Case Generation"IJIET, (2014).

[7] Singh Naveen, Mrs. Kavita Agarwal "Software Testing using Evolutionary Approach", IJSR2013

[8] Tandon Anisha, Poonam Malik "Breeding Software test cases with genetic algorithm" IJAET, (2012).

[9] Windisch Andreas, Stefan Wappler et al (2007)" Applying Particle Swarm Optimization to Software Testing "ACM.

[10] Yang Xin-She (2014)" Swarm Intelligence Based Algorithms: A Critical Analysis".

 ISSN: 2321-8169
109 – 114

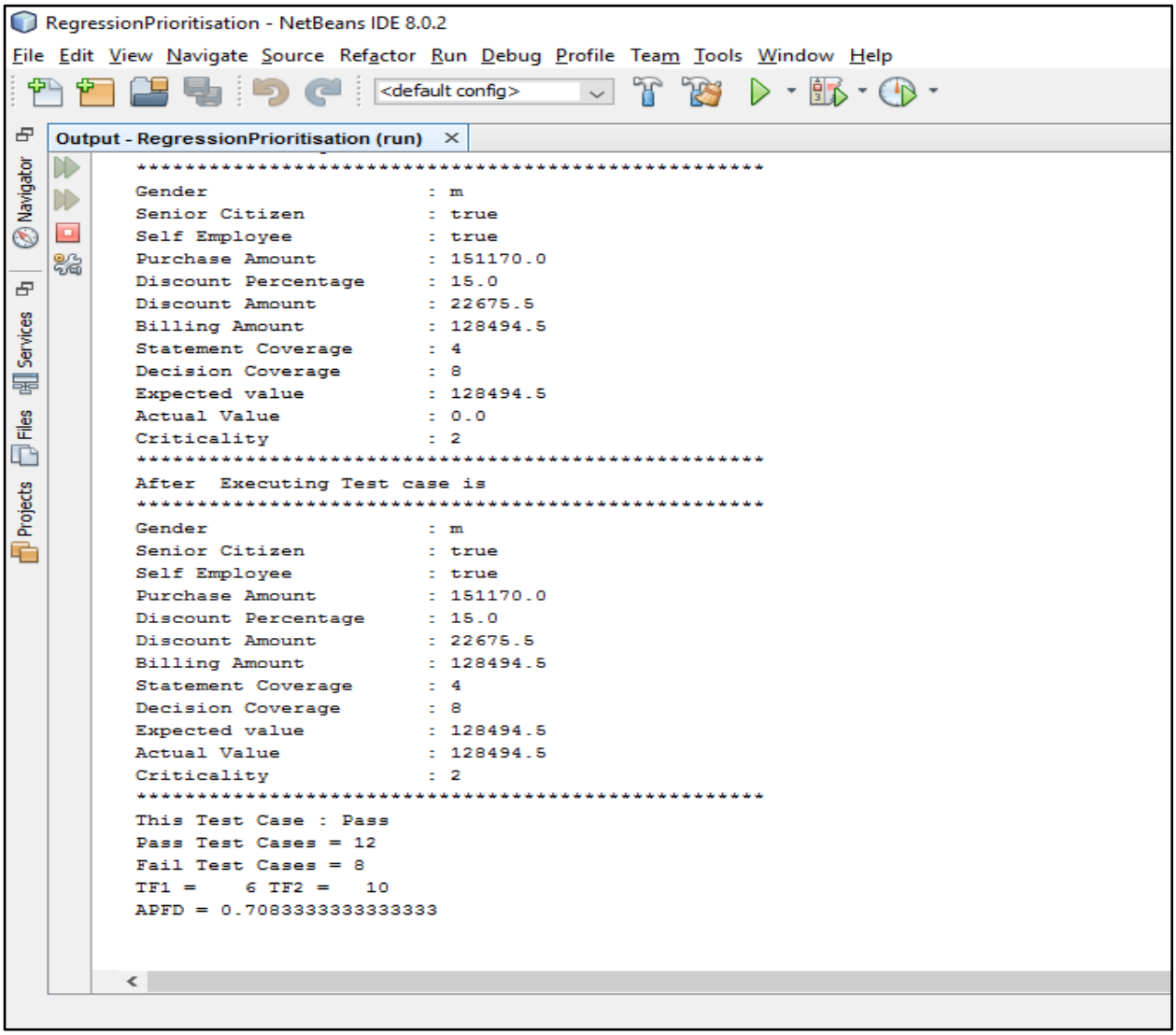


Figure 1. A snapshot of APFD for Un-prioritized Regression Test Suite (URTS)

 ISSN: 2321-8169
109 – 114

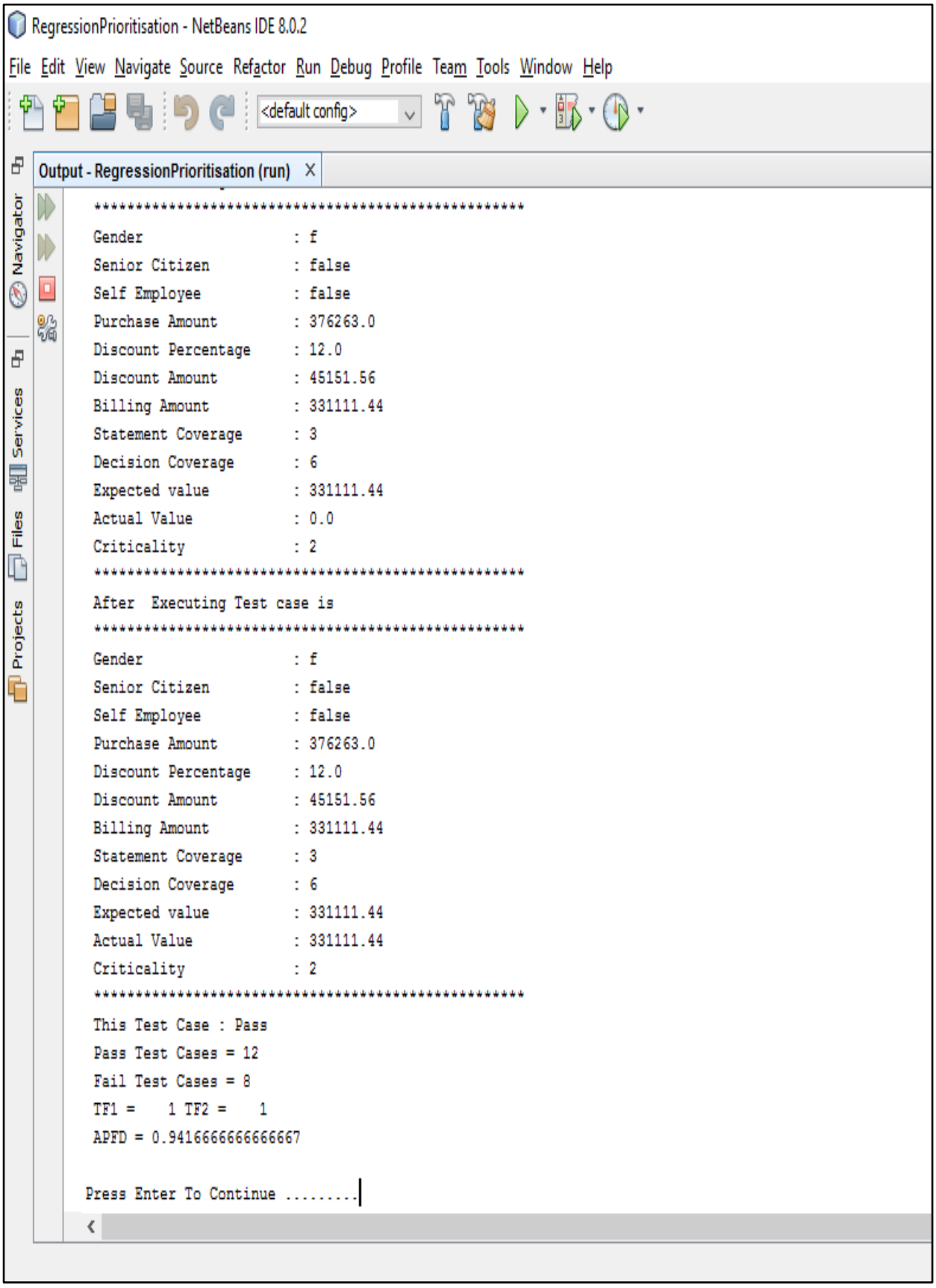


Figure 2. Asnapshot of APFD for Prioritized Regression Test Suite (PRTS)

ISSN: 2321-8169
109 – 114

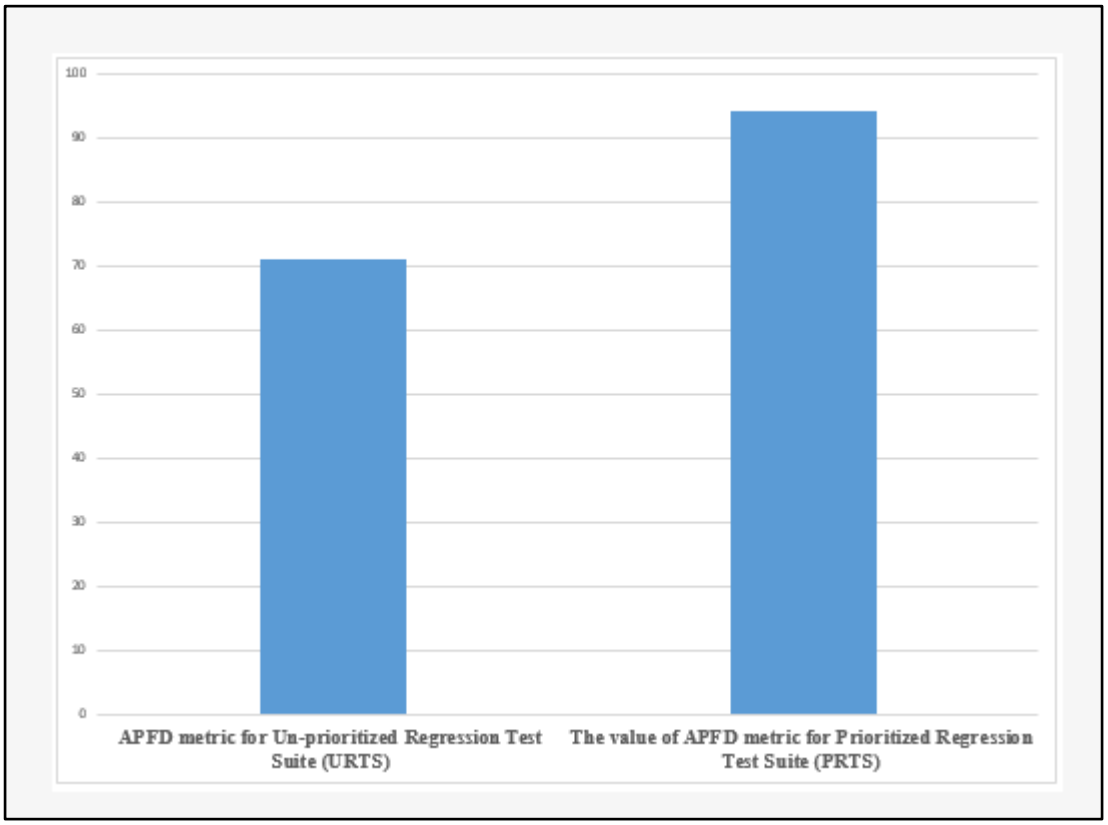


Figure 3. Comparison of APFD for Un-prioritized Regression Test Suite (URTS) and Prioritized Regression Test Suite (PRTS)