International Journal of Engineering and Advanced Technology (IJEAT)
ISSN: 2249-8958 (Online), Volume-8 Issue-2S2, January 2019

Lrpt- Linear Regression Based Prioritization Techniques for Early Fault Detection

John Bruce. E, T. Sasi Prabha

ABSTRACT: ***The role of Regression testing in the maintenance of software systems is significant when it undergoes frequent revision and enhancements. A novel technique for test case prioritization has been proposed, in this paper. The objective is to maximize fault coverage and to improve the efficiency. The test cases are ordered and analyzed with brute force prioritization techniques with initial, reverse and random ordering and Linear Regression based prioritization technique (LRPT). In this technique, regression models are used as a basis for selecting the features form as the components of the selection of test suite. The finely selected order of test suite prioritizes test cases for maximum fault coverage with fewer execution of test suite and its effective is compared with brute force orderings. The results of LRPT method shows an increase in the percentage of faults detected in comparison with brute force approaches.***

KEYWORDS: ***Linear Regression, Prioritization, Fault Detection, Data Reduction, Coverage Criterion etc.***

I. INTRODUCTION

In testing process the adaptation done in one of the component do not affect remaining ones is the objective of Regression model. Software is constantly modified; either in fixing bugs or by addition of new functionality. In general 80% of the software post implementation cost is consumed by Regression testing and hence the most significant challenges of the testing team is to optimize the testing cost. According to a survey, developers makes changes in 20 and more lines of coding per minute which ultimately results in the execution of test cases up to 100 million. More modification results in larger test suites and may execute for days, months or weeks. Automatic test case prioritization is adopted to resolve this issue. It is mandatory for the software team to give importance to the test cases for the earlier detection of faults. The Testing manager assigns weights as critical, unimportant and insignificant on its severity. A number of prioritization techniques like Greedy method, Meta-heuristic method, Data Mining and Machine learning techniques are available in the literature. All these methods functions based on the prior knowledge of the ordering of the test cases. Regression testing works under the ordering of analysis: based on policy reporting, based on components of code that are not detected earlier and based on the earlier information to detect faults faster. The approach which proposed in this paper is focused towards the ordering of test cases intended for the earlier fault detection, is compared with Initial Ordering, Reverse ordering, Linear Regression based Prioritization.

II. PROBLEM STATEMENT

During the process of software development and testing the generated deliverables and the enhancements or the changes made to the existing software are validated according to the changes in the dynamic environment. A test suite will also have to be changed now and then and it is challenging for testers to handle this situation. The most important features of Regression Testing are Selection of Test cases, reducing them and prioritizing them. The goal of software selection deals by the identification of selected test cases which satisfies the constraints, where as Test case reduction produces a much economic Test suite. Test case Prioritization can reorder the test cases selected through Test case selection. The challenge lies in producing it are the cases that provide the same requirement coverage as the given suite with minimum cost. And the importance lie on the responsibility of detecting failures in the test suites that are considered.

In Prioritization of test cases intended to work for coverage of code without considering the detection of faults may produce some undesirable results. The definition of test cases varies as the objective of users varies from time to time. For a program P and a Test case T, a set of test suites those are executed by the same test suite is said to cover the former. This strategy results in the generation of system exposure and error discovery. It is required to quantify the effectiveness of the validation in a quantifiable manner, for which the metric namely Average Percentage of Fault Detection (APFD) has been in usage widely. The values of APFD ranges between 0 and 100 and the rate of error finding increases with the increase in the APFD value. Considering a test suite as T having a set of test cases, g is number of faults in program under test. TFi denotes the index of the initial case covering a given ith fault, where i ≤ m then APFD is calculated as shown in equation (1).

$$APFD(T,P) = 1 - \frac{\sum_{i=1}^{g} reveal(i,T)}{ng} + \frac{1}{2n} \quad \cdots\cdots (1)$$

Manuscript published on 30 January 2019.
* Correspondence Author (s)
John Bruce. E, Research Scholar, School of Computing, Sathyabama Institute of Science and Technology (johnbruce@sathyabama.ac.in)
Dr. T. Sasi Prabha, Professor, School of Computing, Sathyabama Institute of Science and Technology (tsasipraba@yahoo.com)

III. LITERATURE SURVEY

Clustering of requirements engineering into the software testing phase will result in the earlier detection of fault[1]. Reordering of test cases can greatly influence the earlier detection of faults [2]. Prioritizing test cases plays a vital role when there are not enough resources to support testing.

Chen L. et. al has proposed a regression testing technique which deals with dependence analysis. Initially, dependency relationship was analyzed by the authors using the control and data flow information. In the next step, a weighted graph is constructed by the authors. Based on these weights and those which is having highest modifying elements, tests cases are prioritized [3].

Chin-Yu Huang et al (2010) has proposed a GUI based weighted score method for ranking the events. The critical event will be assigned with higher weight. If the weight is large, many faults can be found by running this test case. It has been proved that fault detection can be achieved by the adoption of adjusted-weight ordering [4]. Dan Hao et al (2016), has proposed a Greedy method for the identification of code coverage. The problem of prioritization is represented as linear programming (ILP). A pre-condition is set to implement the prioritization which deals with the various coverage problems. The experimental results have proved that that coverage-based technique has shown fault detection compared with the brute force approaches[2][5][8].

Dianxiang Xu et al (2015) has proposed an automated technique (MISTA) aimed to improving the functionality and security testing of software[6]. Dobolyi et al has done testing with Web based applications [7].Gregg Rothermel and Roland H. Untch (2001), have developed techniques for ordering test cases based on coverage of code for early fault detection. Test cases are assigned with priorities based on the prior knowledge of the testers. An efficient rate of fault detection can help debugging faster [9]. Joao Felipe Silva et al has used parameters namely General Techniques for Prioritization, Branch statements and Loops in order to be generate test cases for early detection of faults [10].

Manika Tyagi and Sona Malhotra (2015) have used Test Case Ranking formula based on which prioritization of test cases is done [11]. Miso Yoon et al (2012) and Usha Badhera et al (2012) have defined a regression based function for prioritizing test cases by the reduction of test suites [12][16]. Test case prioritization was done with the help of Genetic algorithm. Authors have focused on implementation of the test suite which is generated with for the supporting regression testing[7]. The proposed prioritization technique deals with system level testing for efficient fault detection rate [13] [14] [22].Sepehr Eghbali and Ladan Tahvildari (2016) has demonstrated Coverage based technique at Method level and Block Level by Generalized AT using Lexicographical Ordering for breaking ties [15]. Vaithyasubramanian. S et al. proposed several techniques for Password and ATM PIN generation [17 - 20], Wang et al analyzed (2107) analyzed 97.2 million Passwords using linear regression model to analyze policies and protocols [21].

Software maintenance is an activity which includes continual enhancement, fault detection and optimization of significant features. These changes may in turn affect the performance of the system. Regression testing is thus adopted at this stage in order make sure that the system is does not fail adversely. Regression testing has been estimated to cover up to 50% of the overall software maintenance cost, Dianxiang Xu et al (2015) [6]. Regression testing with various Test suites is a laborious and tedious task. Optimization of test suites to a minimal subset by adopting coverage criterion can improve the overall software maintenance cost. Approaches using Prioritization of Test cases, Distance Measure, Greedy algorithm, Search algorithms, Meta heuristics algorithms, Hill Climbing method, Ant colony optimization and Genetic algorithms are existing for prioritization of test cases. These strategies are is more applicable for code-based contexts.

IV. PROPOSED METHODOLOGY

The methods proposed in literature are mainly on statistical, dictionaries or knowledge-based approaches. Good ordering of test cases helps to reveal high risks which there by improves the reliability of the software under test. Given a set of training data, a statistical model (Parametric or Non-parametric) can be used to estimate f. Linear Regression is adopted for finding linear relationship between independent variables and one dependent variable. Linear Regression states the existence of relationship between the input and the output data.

A model can be represented as :$Y = f(X_1, X_2, X_3) + \varepsilon$, where f is a specified function and ε is the error. intercept The parameters in the Linear Regression models are easy to interpret, where in β_0 is the intercept and β_j is the slope of the jth variable X_j.

In Non Parametric models such as the response and predictor variables need not to be specified as they can be fitted with large number of functional forms. The main challenges in these models deal with:

Finding the type of relationship between Y's and X's?

Which particular predictors affect the response?

Is the relationship positive or negative?

Is the relationship a single linear or a complicated one?

Consider Vectors a = [a(1), ….a(k)] ε Fk and b = [b(1), …b(k)] ε Fk , where F is the set of real numbers and x is said to have a higher precedence than y, shown as a > b or b < a if considering the features as vector components. ∃ i > 0, ∋ j (f, a(j) = b(j) and a(i) > b(i) For eg, Consider the following 3 Test cases :a = [1,2,3,7], b = [1,2,3,7,14], c= [1,2,3,7,14]. The vectors can be ordered as b > c > a.

4.1 Residual Analysis

Two major components of regression models are Randomness and Unpredictability. Prediction is measured as the sum of Deterministic (predictor variable) and Statistical measures. There will be some information, that are missed to cover and these can be obtained from the residual information. Residual plot helps in analyzing the model using the residual values and it is plotted between predicted and residual values.

Published By:
Blue Eyes Intelligence Engineering and Sciences Publication (BEIESP)

The distance of the point from the origin specifies how poor the prediction is. If the value is positive, the prediction is low and if the value is negative, the prediction is high. A perfect prediction is where the value is 0. The model can be improved by predicting detecting residual patterns.

4.2 Metrics for Evaluation of the Model

R-squared value : R-squared value ranges between 0 and 1.

1. Regression Sum of Squares (SSR) : SSR depicts how far estimated regression is far from the horizontal 'no relationship' line.

$$SSR = \sum(\hat{y} - \bar{y})^2 \quad \ldots\ldots(2)$$

2. Sum of Squared Error (SSE) : SSE indicates how far the target value varies around the regression line.

$$SSE = \sum(y - \hat{y})^2 \quad \ldots\ldots(3)$$

3. Total Sum of Squares (SST) : SST indicates how far the data points are centered around the mean.

$$SST = \sum(y - \bar{y})^2 \quad \ldots\ldots(4)$$

When variables are included in the model, values of SSE decreases whereas values of SSR increases. Larger values of the coefficients of determination R^2 = SSR/SST demonstrates a better fitting model.

4.3 EFFECTIVENESS OF PRIORITIZED TEST SUITE

In order to quantify the rate of fault detection of a test suite, a weighted average of the percentage of faults detected (APFD) is used. For instance, consider a program with 10 different versions and a test suite of 7 test cases : F1 to F7. Table 1, shows the various test cases exposed by the usage of the test suites. Three control techniques for ordering of test cases are adopted as discussed below:

(i) Initial ordering: Faults does not exists equally in each function. Start with the faults which are known and identify which fault can a test case expose. Order the test cases that maximize the rate of faults identified according to their assigned order. Thus, these test cases are prioritized based on their previous history.

(ii) Reverse ordering : Test cases are ordered in reverse (from last test case to first test case)

(iii) Ordering based on Priority (LRPT) :Linear regression is applied and the combinations of test cases that are highly correlated are obtained. A dependent variable is predicted from several independent variables. A linear combination of these variables that best predict an outcome is to be analyzed and retrieved.

Table 1 : Test Suites with faults exposed

	TC1	TC2	TC3	TC4	TC5	TC6	TC7	TC8	TC9	TC10
FT1	√	√								
FT2		√						√		
FT3				√					√	
FT4				√						
FT5	√				√	√	√			
FT6							√	√		√
FT7	√						√			
# defects	3	2	0	2	1	1	3	2	1	1
Time	6	2	0	3	2	2	9	3	2	1
severity	1	1	1	2	2	2	1	2	2	3
RFT	5	5	0	6.67	5	5	3.33	6.67	5	10
RFD	0.97	.32	0	0.65	0.32	0.32	0.97	0.65	0.32	0.32
RDA	0.5	0.5	0	1.33	1	1	.33	1.33	1	3
TCR	6.47	5.82	0	8.65	6.32	6.32	4.63	8.65	6.32	13.32

(i) Initial ordering

$$TP1 = 1 - \frac{(1+2+4+4+1+7+1)}{7 * 10} + \frac{1}{2 * 10} = 0.6642$$

(ii) Reverse ordering

$$TP2 = 1 - \frac{(9+3+2+7+4+1+4)}{7 * 10} + \frac{1}{2 * 10} = 0.464$$

(iii)Ordering based on Priority: T1 , T4, T2, T5, T6, T7, T3, T9, T10 and T8

$$TP3 = 1 - \frac{(1+3+2+2+1+0+1)}{7 * 10} + \frac{1}{2 * 10} = 0.807143$$

RFT – Rate of fault detection

$$RFT_j = \left[\frac{RFTj}{Timej}\right] * 10 \quad \ldots\ldots..(1)$$

PFD = percentage of fault detected

$$PFD = \left[\frac{Nj}{N}\right] * 10 \qquad (2)$$

Risk Detection Ability

$$RDA_j = \frac{(Sj * Nj)}{timej} \quad \ldots\ldots..(3)$$

4.4 The number of test cases that fail in the test suite

The objective of this work lies in the earlier detection of faults For testing purpose, the online Flipcart website is taken.

Published By:
Blue Eyes Intelligence Engineering
and Sciences Publication (BEIESP)

The APFD measure depends upon two important assumptions: All faults have same weightage and as they have same computational cost. This assumption may not reveal results always.

V. RESULTS AND DISCUSSIONS

This research work answers two basic questions: (i) How can prioritization enhance fault detection and (ii) How does the predictions of occurrences of faults better the fault detection ? For answering these questions, the website of Flipkart is used. Testing is done for four functionalities namely login, search, direct search, change options, details, checkout and cancellation and logout

Taking all the seven functionalities, the residual standard error was notified at 17.2 on 819 degrees of freedom, with p value being 0.4054 indicating no significance among the independent variables. The representation is depicted in fig 1, in which the values are under fitted. By repeated iterations, the residual standard error is 0.4554 on 67 degrees of freedom in which 29 observations deleted due to missingness and the p-value is 0.004726, in which the irrelevant features are removed and only the required features are retrieved and the representation is given in fig.2, showing the observed values overfitting the model. The scatter plots for the reduced sets of data are represented in figures 3,4,5 and 6 and it shows order 2, order 3 and order 4 grows linearly whereas order 1 is quadratic in nature with order 2, order 3 and order 4 being identified as the optimum orderings. The test suites thus obtained are the ones which satisfies the maximum coverage criterion.

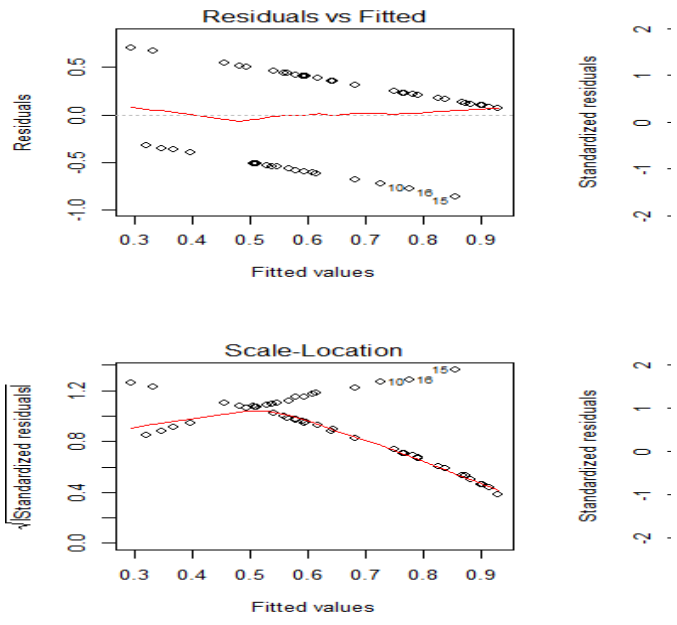

Fig. 1 Standard residual errors before prioritization

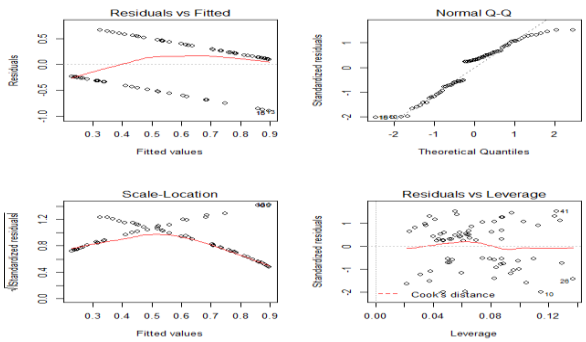

Fig. 2 Standard residual errors after prioritization

Suppose if the test cases are arranged in the sequence TC1,TC2,…TC10 in order to form a test suites. This prioritization is done based on the past history. Figure 3 show the faults identified in percentage versus the test suite TC1 used. After running test case TC1, three among 7 faults are identified.; thus 42.85% faults are detected after 0.3 of the test suite TC1 is used. Execution of TC2, one more fault is identified where increasing the fault detection to 57.14%. After executing TC3, remains the same. Execution of TC4 produces 85.71% of the faults and finally starting from TC6, execution of test cases produces 100% of code coverage which is depicted in TC1. APFD scores for prioritized test suite TC2 is depicted in Fig. 4 whereas the APFD scores for LRPT based prioritization is depicted in Fig. 5 and the comparison of initial ordering, reverse ordering and LRPT ordering is depicted in Fig. 6.

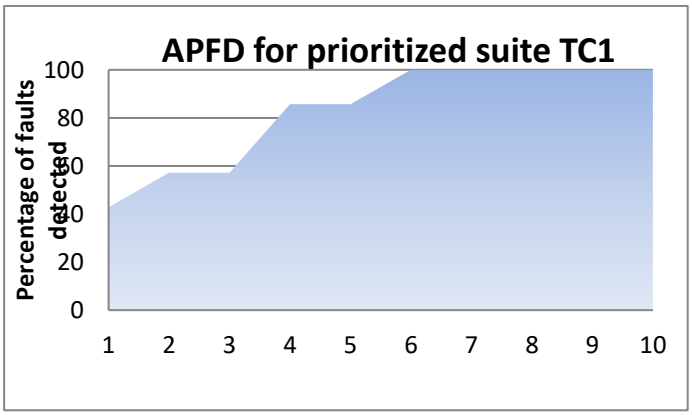

Fig 3 APFD for test suite 1

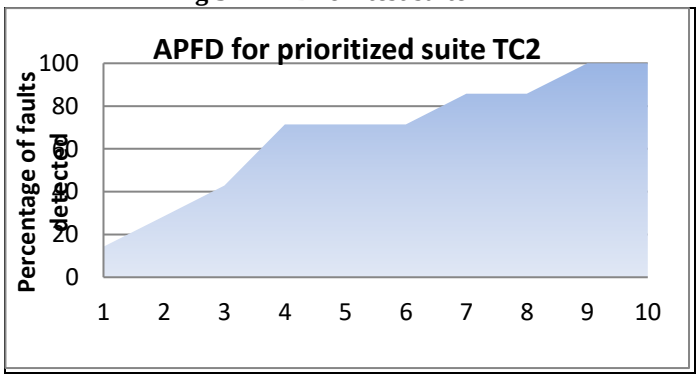

Fig 4 APFD for test suite 2

Published By:
Blue Eyes Intelligence Engineering
and Sciences Publication (BEIESP)

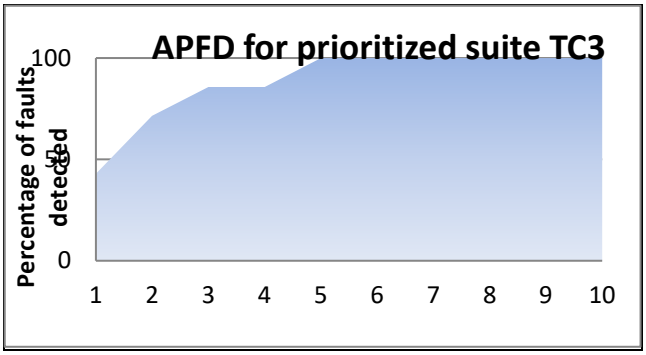

Fig 5 APFD for test suite 3

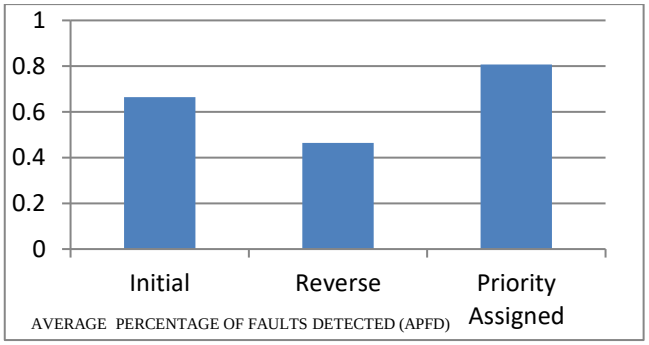

Fig 6 Comparison between APFD scores

VI. CONCLUSION

Experimental results have shown LRPT approach has shown maximum coverage of faults than existing orderings. The results which occur for functional level are more expensive than statement level techniques. The rate of fault detection is not of paramount important, when the size of test suite is short. When the time taken to execute all test cases are high, these differences may be significant. APFD, the measure of fault detection, captures the effectiveness of prioritization only partially. Linear Regression based Prioritization Technique is adopted in this paper for maintaining "Optimal" orderings. This algorithm not always fetches optimal ordering but shows some heuristics which outperform the outliers produced by optimal ordering.

REFERENCES

1 M. J. Arafeen and H. Do, "Test case prioritization using requirements-based clustering," in Software Testing, Verification and Validation (ICST), 2013 IEEE Sixth International Conference on, pp. 312–321, IEEE, 2013

2 AndreeaVescan, Camelia Serban, Camelia Chisalita-Cretu, Laura Diosan (2017), "Requirements Dependencies –based Formal Approach for Test Case Prioritization in Regression Testing",13th IEEE International Conference on Intelligent Computer Communication and processing

3 Chen, L., Wang, Z., Xu, L., Lu, H., & Xu, B. "Test case prioritization for web service regression testing", Service Oriented System Engineering (SOSE), Fifth IEEE International Symposium (pp. 173-178). 2010

4 Chin-Yu Huang, Jun-Ru Chang and Yung-Hsin Chang (2010), "Design and analysis of GUI test-case prioritization using weight-based methods", The Journal of Systems and Software, Elsevier, 83(2010), Pp. 646-659

5 Dan Hao, Lu Zhang, Lei Zang, Yanbo Wang, Xingxia Wu, and Tao Xie (2016), "To Be Optimal or Not in Test-Case Prioritization", IEEE TRANSACTIONS ON SOFTWARE ENGINEERING, 42(5), Pp. 490-504 .

6 Dianxiang Xu, Weifeng Xu, Michael Kent, Lijo Thomas, and Linzhang Wang (2015), "An Automated Test Generation Technique for Software Quality Assurance", IEEE TRANSACTIONS ON RELIABILITY, 64(1), Pp. 247 -268

7 Dobolyi, K., Soechting, E., & Weimer, W. (2011). Automating regression testing using web-based application similarities. International Journal on Software Tools for Technology Transfer, 13(2), 111-129

8 Elbaum, Sebastian; Malishevsky, Alexey G.; and Rothermel, Gregg, "Prioritizing Test Cases for Regression Testing" (2000). CSE Technical reports. 27. http://digitalcommons.unl.edu/csetechreports/27

9 Gregg Rothermel, Roland H. Untch, Chengyun Chu and Mary Jean Harrold (2001), "Prioritizing Test Cases For Regression Testing", CSE Journal Articles, IEEE Transactions on Software Engineering, 27(10), Pp. 929 – 948.

10 Joao Felipe Silva Ouriques, Ernanuela Gadelha Cartaxo and Patricia Duarte Lima Machado(2015), "Revealing influence of model structure and test case profile on the prioritization of test cases in the context of model-bases testing", Journal of Software Engineering Research and Development, 3(1), Pp. 1-28, DOI 10.1186/s40411-014-0015-5

11 Manika Tyagi and Sona Malhotra (2015), "An Approach for Test Case Prioritization Based on Three Factors", I.J. Information Technology and Computer Science, Pp. 79-86

12 Miso Yoon, Eunyoung Lee, Milkyoung Sung, Byoungju Choi (2012), "A Test Case prioritization through correlation of requirement and risk", Journal of Software Engineering and Research, Vol. 5, Pp. 823-835.

13 Raju, S., and G. V. Uma. "Factors oriented test case prioritization technique in regression testing using genetic algorithm." European Journal of Scientific Research ,pp. 389-402,2012

14 Roland H. Untch and Gregg Rothermel (2001), "Prioritizing Test Cases For Regression Testing", CSE Journal Articles, 27(10), Pp. 929 – 946.

15 Sepehr Eghbali and Ladan Tahvildari (2016), "Test Case Prioritization Using Lexicographical Ordering", IEEE TRANSACTIONS ON SOFTWARE ENGINEERING,Vol. 42 (12), Pp. 1178 – 1195

16 Usha Badhera , G.N Purohit , Debarupa Biswas (2012)," TEST CASE PRIORITIZATION ALGORITHM BASED UPON MODIFIED CODE COVERAGE IN REGRESSION TESTING", International Journal of Software Engineering & Applications (IJSEA), 3(6), Pp. 29-37

17 Vaithyasubramanian, S., & Christy, A. (2015). A scheme to create secured random password using Markov chain. In Artificial Intelligence and Evolutionary Algorithms in Engineering Systems (pp. 809-814). Springer, New Delhi.

18 Vaithyasubramanian, S., Christy, A. and Lalitha, D., 2015. Generation of array passwords using Petri net for effective network and information security. In Intelligent Computing, Communication and Devices (pp. 189-200). Springer, New Delhi.

19 Vaithyasubramanian, S., Christy, A. and Lalitha, D., 2015. Two factor authentication for secured login using array password engender by Petri net. Procedia Computer Science, 48, pp.313-318.

20 Vaithyasubramanian, S., & Christy, A., 2019. ATM PIN generation - a formal mathematical model to generate PIN using regular grammar, context free grammar and recognition through finite state machine, pushdown automata. Inderscience, International Journal of Internet Protocol Technology, Vol.12, No.1, pp.11-15, doi: 10.1504/IJIPT.2019.098485.

21 Wang D, Cheng H, Wang P, Huang X, Jian G. Zipf's law in passwords. IEEE Transactions on Information Forensics and Security. 2017 Nov;12(11):2776-91.

22 S. Yoo and M. Harman (2012) , "Regression Testing Minimization, Selection and Prioritization: A survey, "Software Testing, Verification and Reliability, 22(9), Pp. 67-120, doi: 10.1016/j.j.indsof.2008.08.00.

Published By:
Blue Eyes Intelligence Engineering and Sciences Publication (BEIESP)