Improving Fault Detection Rate using Similarity-based Test Case Prioritization in Regression Testing

Shilpi Singh, Preetish Ranjan, Raj Shree

Abstract: ***Rapid evolution in software requires regression testing to be performed as an essential activity which validates the software before the next release. Where software developer may add or removes intended features to maintain the software according to the customer requirements. In that case, complete test cases execution is nearly infeasible due to limited time and resources. So, the main aim of prioritization is to test any software with minimal time and maximum efficiency in terms of fault coverage rate. This paper proposed different similarity-based prioritization techniques to provide ranking to the test cases based on their influence level which is computed as similarity degree in three levels for the software to be tested. Each level represents the integration of selected coverage criteria's. In order to validate our proposed technique, we have conducted a case study to measure its effectiveness in prioritizing the test cases. We experimentally observed that by incorporating a similarity-based approach with more than one coverage criteria; results for similarity-based prioritization are promising than any other conventional coverage based approaches in terms of Average Percentage of Faults Detected.***

Keywords ***: Software Testing, Regression Testing, Test case prioritization, APFD.***

I. INTRODUCTION

Regression testing ensures that the enhancement of any software doesn't affect the performance of the previous one. This is a very complex activity in perspective of time and cost where more than 50% of the cost is used in the software maintenance phase [4]. In this competitive world, software changes rapidly due to frequent advancement required by the customers. In that case, testing requires a number of existing and new test cases to be executed to reveal maximum faults after certain modifications. But to execute a complete test is impractical within limited time and resources. To solve the above problem researchers classified test suite optimization into three main categories i.e.

Revised Manuscript Received on November 30, 2019.
***** Correspondence Author

Shilpi Singh*, Amity School of Engineering and Technology, Amity University Patna, Patna, India.. Email: shilpi.singh.it@gmail.com
Preetish Ranjan, Amity School of Engineering and Technology, Amity University Patna, Patna, India.. Email: mail.preetishranjan@gmail.com
Raj Shree, Department of Information Technology, Babasaheb Bhimrao Ambedkar University, Lucknow, India.. Email: rajshree.bbau2009@gmail.com

TCS (test case selection), TSM (test suite minimization) and TCP (test case prioritization). Several test suite optimization techniques have been proposed to solve the above problem. Test case prioritization is one of the important techniques for ranking the test cases with respect to selected testing criteria. Selection of testing criteria depends on several goals, such as (1) the test cases with higher coverage are executed earlier (2) to increase the fault detection ability (3) to decrease the time and cost required for satisfying the selected coverage goals [1, 5]. The aim of any prioritization technique is to increase the possibility of meeting certain criteria by the resultant test suite with earlier fault detection ability. Coverage-based test case prioritization primarily concentrates on different code coverage information to rearranges the order of test cases to reveal maximum faults as early as possible [1]. However, previous studies show that in some cases, code coverage as selection criteria is not sufficient to guarantee a high fault detection rate [6].

In recent years, similarity-based test case prioritization techniques, combining clustering approach [7], ART-approach [8] and other similarity-based prioritization [9], have gained more attention. Similarity-based test case prioritization techniques primarily concentrated on test case diversity that helps to detect more faults [10, 11]. But, most of the techniques use single criteria to measure the diversity between test case pairs that are not sufficient to get an optimal result. However, the optimal test suite is best generated by multiple coverage criteria [12, 13, 15, 16, 18].

In this paper, we present several similarity-based prioritization techniques based on pair-wise selection strategy. Pair wise comparison of test cases is a fundamental strategy to inspect test cases and choose an association between them in a finite test set. Our technique evaluates the similarity degree for each test case pair in three levels and accordingly assigns the execution order to the test cases in a test suite. We also empirically evaluate their improvement in fault detection rate that let developers initiate debugging and amending faults before the release of any software product. The results show that some of the proposed techniques can attain higher fault detection rate, in terms of APFD, in comparison with the other considered techniques and random ordering. The rest of the paper is organized as follows. Section 2 illustrates the background of the proposed work. After that, we presented our proposed test case prioritization techniques in Section 3, followed by an experimental evaluation in Section 4.

Finally, we conclude this paper in Section 5.

II. BACKGROUND

A. Test Case Prioritization

Test case prioritization is one of the regression testing techniques widely used for getting an optimal test suite. The technique reassures the high fault detection rate of a test suite by executing essential test cases on priority basis. That results in earlier detection of faults as well as earlier initiation of debugging process. The test case prioritization problem is framed as follows [1]:

Given: a test suite TS; the set of permutations PT of TS; and, a function f from PT to the set of all real numbers.

Problem: To find TS'$\in$PT such that,

$\forall$TS''$\in$ PT $\Rightarrow$ f (TS') $\geq$ f (TS'')

Coverage Similarity

The test case pair is said to be similar, if they are identical in terms of some selected criteria, where, the criteria may be coverage also. Coverage might be a branch, statement, def-use, control-flow, MC/DC, function, path, and data flow etc. We can apply more than one criterion to measure the similarity value, which represents how much the test case pair is identical to each other [12, 15]. With the help of coverage similarity, any prioritization technique can improve the ordering of test cases which may also increase the fault detection rate.

III. PROPOSED APPROACH

Here we introduced the proposed approach in detail which is primarily divided into four phases:

Instrumentation: Software under test is executed on predefined environment to gather test case coverage detail (statement, MC/DC, and branch coverage) for test suite.

Distance calculation: The distance between test case pair for each criterion are calculated.

Integrated similarity degree calculation: The similarity values of test case pairs for each criterion are integrated.

Apply prioritization techniques and calculate APFD: Apply techniques to order the test cases and then evaluate fault detection effectiveness of prioritized test suite.

A. Test Case Similarity

The objective of the similarity based test suite prioritization strategy is to order the test cases based on the similarity values among them. For similarity analysis, certain distance measures are available which helps to evaluate the difference level between the test case pair [17]. The resultant ordered test suite must satisfy the coverage requirements as the original one. Hence, the goal is to achieve maximum requirement coverage and higher fault detection rate by the ordered test cases.

Cartaxo et al. [14] proposed a similarity function as a comparison metric to find the most similar and diverse test cases in a test suite.

$$Similarity\ function\,[i,j] = \ nit\,/Avg\,(|i|,|j|) \qquad (1)$$

Where nit represents the number of identical transition and (|i|, |j|) represents the average transition between path lengths. With the help of the calculated similarity degree of each pair of test cases the $n \times n$ square similarity matrix is created, where n is the number of paths and each path represented as i ($1 \leq i \geq n$) is called a test case.

Coverage: Coverage of any test case (TC) is a set of three coverage criteria$\{\langle s_1, m_1, b_1\rangle, \langle s_2, m_2, b_2\rangle, \ldots, \langle s_n, m_n, b_n\rangle\}$, where $\langle s_i, m_i, b_i\rangle$ signifies a statement coverage specification s_i, modified decision/condition coverage (MC/DC) m_i, and a branch specification b_i. Let $(T) = \{s_1, s_2, \ldots, s_n\}$, $M(T) = \{m_1, m_2, \ldots, m_n\}$, and $B(T) = \{b_1, b_2, \ldots, b_n\}$, signify the set of statement coverage specifications, the set of MC/DC specification, and the set of branch specification, exercised in the execution of test case t to validate the software to be tested, respectively. Here coverage set may be of more than three coverage criteria which absolutely depend on user's choice.

Coverage Similarity: We presented step-wise procedure of evaluating similarity degree between two test cases T_i and T_j i.e. (Step-i) Statement similarity specification (CS-I), (Step-ii) MC/DC similarity specification and statement similarity specification (CS-II), and (Step-iii) Branch, statement, and MC/DC similarity specification (CS-III).

Let the Coverage Set of any test case pair T_i and T_j be $\langle S_i, M_i, B_i\rangle$ and $\langle S_j, M_j, B_j\rangle$ respectively. Let the distance between statement, MC/DC and branch coverage of the test case pair T_i and T_j be $\pi(S_i, S_j)$, $\pi(M_i, M_j)$ and $\pi(B_i, B_j)$. Where, the distance is calculated by applying Eq.1. The proposed techniques use the 'Geometric Mean' to evaluate the integrated similarity degree for test cases T_i and T_j, which is calculated in three steps:

Step 1:

$$CS - I = \ \pi(S_i, S_j\,) \qquad (2)$$

Step 2:

$$CS - II = \sqrt[2]{\pi(S_i, S_j) \times \pi(M_i, M_j)} \qquad (3)$$

Step 3:

$$CS - II = \sqrt[3]{\pi(S_i, S_j) \times \pi(M_i, M_j) \times \pi(B_i, B_j)} \qquad (4)$$

However, we demonstrate our techniques using three criteria, but it is also flexible to simplify the above equations to function on more than three criteria.

B. Test Case Prioritization

Our proposed technique uses similarity degree between each pair of test cases to rank the test cases.

Suppose we have a test suite 'TS' of 'm' number of test cases. Select all possible pairs of different test cases and divide them into K different groups. Each group comprising test case pairs with the same similarity value and it is designated by G_k $(1 \leq k \leq K)$, where k represents the group index. A smaller k specifies all test case pairs of that group having maximum similarity value.

The proposed similarity based prioritization techniques are illustrated as follows:

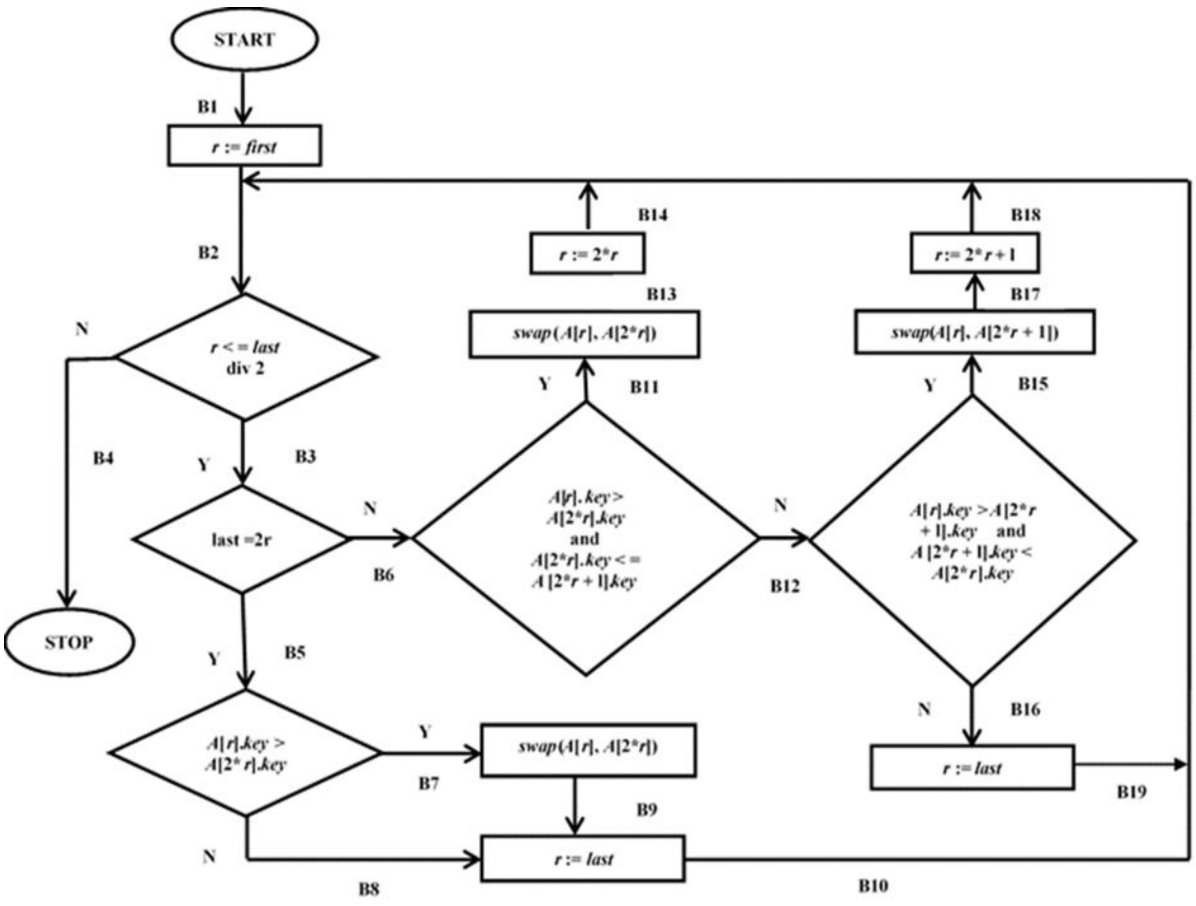

Fig. 1. Control flow diagram of pushdown procedure

PS1 (Total CS-i similarity prioritization): The technique generates K different groups of test case pairs using their similarity values CS-i. At first, a test case pair of maximum similarity value is selected and chooses any one test case T from that pair randomly. Continuously all the pairs containing T were selected from the same group. In case of a tie, the technique arbitrarily picks one pair to break this. Repetitively, the above process is executed for each group one by one until every unique test case has been chosen from each group.

PS2 (Total CS-i dissimilarity prioritization): The technique is similar to PS1, apart from the selection process for test case pair having minimum similarity values, and the above process is executed in ascending order for remaining groups.

PS3 (Iterative CS-i dissimilarity prioritization): In descending order (G_m to G_1) of the group number, the technique selects one test case pair from each group. Excluding the selected pairs, the above selection process repeats on the remaining groups until every test cases have been included for prioritization purpose.

PS4 (Iterative CS-i similarity prioritization): This technique is equivalent to the above technique i.e. PS3, apart from the test pair selection process from each group is in ascending order of the group index (G_1 to G_m) instead of in descending order.

PS5 (Advanced iterative CS-i dissimilarity prioritization): The technique selects one test case pair (T_a, T_b) from group G_m and then the further test case pairs are selected containing T_a or T_b, if available, from each group in descending order of the remaining groups (G_{m-1} to G_1). Afterwards, the selected test case pair from the group is to be removed. For each remaining group, this technique reiterates the above discussed selection process until all the test cases have been included for prioritization purpose.

PS6 (Advanced Iterative CS-i similarity prioritization): This technique is quite similar to the above technique i.e. PS3, apart from that the selection process of test case pair from each group is executed in ascending order of the group index (G_1 to G_m), instead of in descending order.

IV. EXPERIMENTAL DISCUSSION

This section presents the experimentation and evaluation of the proposed prioritization techniques. The experimentation compares the relative performance and effectiveness of the proposed techniques with some standard state-of-the-art prioritization techniques [1].

A. Subject Program

Figure 1 presents a control flow diagram of the standard pushdown procedure [3], a case study on which the proposed techniques have been performed to evaluate the performance. This work also uses hand-seeded faults [2] for the subject program. We have inserted different types of errors in the faulty version of the subject program which is represented by fault matrix (see Table 1). During the evaluation process, three coverage parameters are used i.e. statement, MC/DC, and branch coverage to measure the distance between test cases.

All the mentioned coverage information is calculated manually to validate the proposed techniques. Based on the calculated distance (similarity degree) values, further prioritization will be processed.

B. Effectiveness Measure

The performance metric "average percentage of faults detected" (APFD) introduced by Elbaum et al. [1] has been widely used to evaluate any prioritization techniques. A Higher value of APFD concludes improved fault detection rates.

Let TS be a test suite of n test cases, F represents a set of k faults exposed by test suite, and TF_i signifies that the fault i reveals by the first test case of the prioritized test suite TS'. The equation to calculate APFD value for test suite TS' is as follows:

$$APFD = 1 - \frac{TF_1 + TF_2 + \cdots + TF_m}{nm} + \frac{1}{2n} \qquad (5)$$

C. Experimental Evaluation and Comparison

Before applying any prioritization techniques, for each selected criterion the distance between each pair of test cases must be calculated by using Eq. 1. We evaluate the similarity values of every possible test case pairs in three levels using Eq-2, Eq-3, and Eq-4. We observe from Table 2 that the similarity values between test case pair are relatively different at each level. These values illustrate that use of more than one criterion in similarity-based approach gives the optimal result in comparison to conventional prioritization techniques. Moreover, Table 2 presents the similarity values for few of the test case pairs only. With the help of these coverage similarity values, further prioritization techniques are applied to get ordered test cases.

Table- I: Fault matrix for pushdown procedure

Test Cases	**F1**	**F2**	**F3**	**F4**	**F5**	**F6**	**F7**
T1		×	×				
T2	×	×		×			
T3	×	×		×	×	×	
T4	×					×	
T5		×	×				
T6							
T7		×	×				
T8	×			×			
T9	×					×	
T10	×	×		×	×		
T11		×	×				
T12	×	×				×	×

× **designate fault is revealed, and blank cell indicates fault is not revealed**

We use Table 3 to show the results of different pair-wise test case prioritization techniques (PS1 to PS6) using CS-I similarity values. By considering those similarity values, the different test case pairs of T1 to T12 are categorized into different groups (see leftmost columns of Table 3). By implementing each technique from PS1 – PS6, the possible ordering of test case pairs is shown in the rightmost columns of Table 3. Similarly, all the mentioned prioritization techniques are also applied to CS-II and C-III similarity values.

We also analyze the result of proposed and conventional techniques on the subject program that is presented through Table 4. For each technique, we deliver a valid ordering of test cases in Table 4. For single test case prioritization techniques i.e. S2, S3, and S4, we use three coverage metrics (i.e., Statement, MC/DC, and Branch) to order the test cases. And, for similar test pair prioritization techniques i.e. from PS1 to PS6, we use the coverage similarity metrics i.e. CS-I, CS-II, and CS-III to order the test cases.

Table- II: Name of the Table that justify the values

Test Cases		**T1**	**T2**	**T3**	**T4**	**T5**	**T6**	**T7**	**T8**	**T9**	**T10**	**T11**	**T12**
T1	**CS-I**		0.55	0.25	0.27	0.83	0.33	1.00	0.27	0.50	0.67	0.83	0.60
	CS-II		0.45	0.22	0.24	0.70	0.33	1.00	0.25	0.40	0.61	0.70	0.54
	CS-III		0.45	0.23	0.24	0.69	0.33	1.00	0.25	0.41	0.63	0.69	0.56
T2	**CS-I**			0.55	0.33	0.62	0.40	0.55	0.60	0.54	0.88	0.62	0.50
	CS-II			0.52	0.29	0.59	0.38	0.45	0.60	0.51	0.81	0.59	0.42
	CS-III			0.52	0.29	0.57	0.39	0.45	0.59	0.52	0.79	0.60	0.41
T3	**CS-I**				0.70	0.27	0.50	0.25	0.70	0.63	0.50	0.27	0.58
	CS-II				0.67	0.23	0.40	0.22	0.62	0.61	0.5	0.23	0.58
	CS-III				0.66	0.24	0.41	0.23	0.62	0.60	0.5	0.24	0.57
T4	**CS-I**					0.30	0.75	0.27	0.60	0.70	0.30	0.30	0.63
	CS-II					0.25	0.65	0.24	0.51	0.67	0.28	0.25	0.64
	CS-III					0.26	0.67	0.24	0.53	0.66	0.27	0.26	0.62
T5	**CS-I**						0.37	0.83	0.30	0.55	0.55	1.00	0.50
	CS-II						0.34	0.70	0.27	0.52	0.45	1.00	0.40
	CS-III						0.35	0.69	0.27	0.54	0.45	1.00	0.41

Table- III: Ordering of Test Cases using Coverage Similarities (CS-I)

Similarity Value	Group		Selected test pairs in order						
	Index	Test Case Pairs	Seq. No.	PS1	PS2	PS3	PS4	PS5	PS6
1.00	G1	(T1, T7) (T5, T11)	1	(T1, T7)	(T1, T3)	(T1, T3)	(T1, T7)	(T1, T3)	(T1, T7)
0.90	G2	(T9, T12)	2	(T5, T11)	(T3, T7)	(T1, T4)	(T9,T12)	(T1, T4)	(T1, T5)
0.88	G3	(T2, T10)	3	(T9, T12)	(T1, T4)	(T8, T11)	(T2,T10)	(T1, T6)	(T1, T10)
0.83	G4	(T1, T5) (T1, T11) (T5, T7) (T7, T11)	4	(T2, T10)	(T1, T8)	(T6, T7)	(T1, T5)	(T1, T9)	(T7, T10)
0.75	G5	(T4, T6) (T6, T8)	5	(T1, T5)	(T3, T5)	(T6, T10)	(T4, T6)	(T1, T2)	(T1, T12)
0.70	G6	(T3, T4) (T3, T8) (T4, T9)	6	(T1, T11)	(T3, T11)	(T5, T6)	(T3, T4)	(T3,T12)	(T2, T7)
0.67	G7	(T1, T10)	7	(T5, T7)	(T4, T7)	(T8, T12)	(T1,T10)	(T1,T12)	(T1, T9)
0.66	G8	(T7, T10)	8	(T7, T11)	(T7, T8)	(T2, T6)	(T7,T10)	(T3, T9)	(T1, T6)
0.63	G9	(T3, T9) (T4, T12)	9	(T4, T6)	(T4, T5)	(T8, T9)	(T4,T12)	(T1,T10)	(T1, T4)
0.62	G10	(T2, T5) (T2, T11)	10	(T6, T8)	(T4, T10)		(T2,T11)	(T3, T8)	(T1, T3)
0.60	G11	(T1, T12) (T2, T8) (T4, T8) (T7, T12)	11	(T3, T4)	(T4, T11)		(T2, T8)	(T1, T5)	(T5, T11)

Table- IV: Analysis of Different Prioritization Techniques

Category	Techniques	Index	Order of Test Cases											
			T1	T2	T3	T4	T5	T6	T7	T8	T9	T 10	T 11	T 12
Single Test Case Prioritization Strategy	*Random*	S1	2	4	1	11	8	10	5	3	7	9	12	6
	Total-St Coverage	S2	8	4	1	5	11	9	10	6	2	3	12	7
	Total-MC/DC Coverage	S3	11	5	2	4	9	8	12	7	1	6	10	3
	Total-Br Coverage	S4	9	6	2	5	11	8	10	7	1	4	12	3
Similarity based Test Pair Prioritization Strategy	*Total C-i similarity prioritization (PS1)*	CS-I	1	7	12	9	3	10	2	11	5	8	4	6
	Total C-i dissimilarity prioritization (PS2)	CS-I	1	10	2	4	6	9	3	5	12	8	7	11
	Iterative C-i dissimilarity prioritization (PS3)	CS-I	1	11	2	3	9	6	7	4	12	8	5	10
	Iterative C-i similarity prioritization (PS4)	CS-I	1	5	10	8	7	9	2	12	3	6	11	4
	Advanced iterative C-i dissimilarity prioritization (PS5)	CS-I	1	6	2	3	10	4	11	9	5	8	12	7
	Advanced Iterative C-i similarity prioritization (PS6)	CS-I	1	6	10	9	3	8	2	12	7	4	11	5
	Total C-i similarity prioritization (PS1)	CS-II	1	7	9	10	3	11	2	12	5	8	4	6
	Total C-i dissimilarity prioritization (PS2)	CS-II	1	12	2	6	4	10	3	7	11	8	5	9
	Iterative C-i dissimilarity prioritization (PS3)	CS-II	1	12	2	4	3	11	5	6	10	8	7	9
	Iterative C-i similarity prioritization (PS4)	CS-II	1	5	8	9	7	10	2	11	3	6	12	4
	Advanced iterative C-i dissimilarity prioritization (PS5)	CS-II	1	8	2	4	3	6	12	5	7	9	11	10
	Advanced Iterative C-i similarity prioritization (PS6)	CS-II	1	6	11	10	3	8	2	9	7	4	12	5
	Total C-i similarity prioritization (PS1)	CS-III	1	7	12	11	3	9	2	10	5	8	4	6
	Total C-i dissimilarity prioritization (PS2)	CS-III	1	9	2	4	5	11	3	7	12	8	6	10
	Iterative C-i dissimilarity prioritization (PS3)	CS-III	1	8	2	3	6	10	4	5	12	7	11	9
	Iterative C-i similarity prioritization (PS4)	CS-III	1	5	11	10	9	7	2	8	3	6	12	4
	Advanced iterative C-i dissimilarity prioritization (PS5)	CS-III	1	7	2	3	10	5	11	4	6	8	12	9
	Advanced Iterative C-i similarity prioritization (PS6)	CS-III	1	6	11	10	3	8	2	9	7	4	12	5

To validate the results, we measure the APFD value of the prioritized test suite by each technique using Eq. 5. For example, the resulting APFD's for the three test case orders (Random, Total-St Coverage, and Advanced Iterative CS-i Dissimilarity Prioritization) are 62.56%, 38.62%, and 72.0%, respectively. The test order by PS-5 is, in fact, an optimal order of the test suite in comparison to the test order by S-1 (random) and S-2, ensuring the earliest discovery of the maximum faults. The APFD percentage illustrates that the similarity based prioritization techniques gives a better result as compared to the other conventional techniques.

The analysis of different prioritization techniques is illustrated in Table IV.

The graphical representation of the comparative analysis based on APFD for each subject program is given in Figure 2. The y-axis in the graph represents the APFD percentage against the techniques considered in the x-axis. From Figure 2, it can be concluded that the proposed similar test pair prioritization strategy improved the performance as compared to the random and single test case prioritization strategy. The experimental results indicate that the APFD obtained by applying our proposed approach is on an average greater than that one obtained by applying the random or single test case prioritization strategy. Out of the proposed prioritization techniques, Total C-i dissimilarity prioritization (PS2) and Iterative C-i dissimilarity prioritization (PS3) performs better than other proposed techniques.

V. CONCLUSION

Pair wise selection of test case is a fundamental strategy to compare and prioritize them by measuring their associations between them. Using the idea of similarity approach with multiple criteria is quite attractive to arrange the test cases in some specific order. The main target of this proposed prioritization method is to order the test cases based on the similarity between test cases to conduct regression testing more effectively to achieve more than one performance goals such as statement coverage, branch coverage, and MC/DC coverage. To attain this goal the test cases are prioritized based on level-wise calculated similarity value for each pair of test cases. Using the idea of combining similarity strategy with more than one coverage criteria, we proposed various prioritization techniques to order the test cases based on user's requirement. The proposed work is empirically verified with the help of one standard performance metric i.e. APFD. And results reveal that our techniques are practicable and some of them are quite effective as compared to traditional approaches.

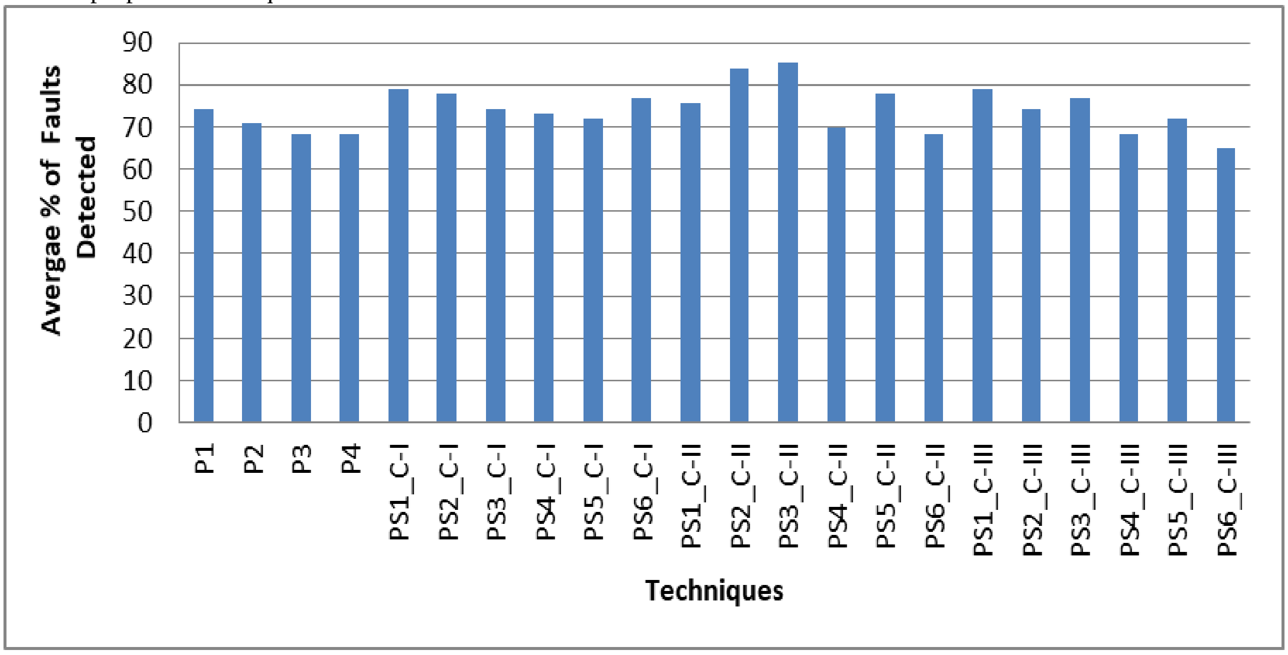

Fig. 2.APFD% based on various prioritization schemes for pushdown program

REFERENCES

1. 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.
2. Andrews, James H., et al. "Using mutation analysis for assessing and comparing testing coverage criteria." *IEEE Transactions on Software Engineering* 32.8 (2006): 608-624.
3. Chen, Tsong Yueh, and Man Fai Lau. "A simulation study on some heuristics for test suite reduction." *Information and Software Technology* 40.13 (1998): 777-787.
4. Harrold, Mary Jean, and Alessandro Orso. "Retesting software during development and maintenance." *2008 Frontiers of Software Maintenance*. IEEE, 2008.
5. Elbaum, Sebastian, Alexey G. Malishevsky, and Gregg Rothermel. *Prioritizing test cases for regression testing*. Vol. 25. No. 5. ACM, 2000.
6. Marick, Brian. *The craft of software testing: subsystem testing including object-based and object-oriented testing*. Prentice-Hall, Inc., 1994.

7. 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.
8. Jiang, Bo, et al. "Adaptive random test case prioritization." *Proceedings of the 2009 IEEE/ACM International Conference on Automated Software Engineering*. IEEE Computer Society, 2009.
9. Ledru, Yves, et al. "Prioritizing test cases with string distances." *Automated Software Engineering* 19.1 (2012): 65-95.
10. Hemmati, Hadi, and Lionel Briand. "An industrial investigation of similarity measures for model-based test case selection." *2010 IEEE 21st International Symposium on Software Reliability Engineering*. IEEE, 2010.
11. Hemmati, Hadi, Andrea Arcuri, and Lionel Briand. "Achieving scalable model-based testing through test case diversity." *ACM Transactions on Software Engineering and Methodology (TOSEM)* 22.1 (2013): 6.
12. Singh, Shilpi, and Raj Shree. "A combined approach to optimize the test suite size in regression testing." *CSI transactions on ICT* 4.2-4 (2016): 73-78.
13. Sharma, Chetna, and Shilpi Singh. "Mechanism for identification of duplicate test cases." *International Conference on Recent Advances and Innovations in Engineering (ICRAIE-2014)*. IEEE, 2014.
14. Cartaxo, Emanuela G., Patrícia DL Machado, and Francisco G. Oliveira Neto. "On the use of a similarity function for test case selection in the context of model-based testing." *Software Testing, Verification and Reliability* 21.2 (2011): 75-100.
15. Singh, Shilpi, and Raj Shree. "A New Similarity-Based Greedy Approach for Generating Effective Test Suite." International Journal of Intelligent Engineering and Systems, Vol.11, No.6, 2018
16. Singh, Shilpi, and Raj Shree. "An Analysis of Test Suite Minimization Techniques." *International Journal of Engineering Sciences and Research Technology* 5 (2016): 252-260.
17. Singh, Shilpi, and Raj Shree. "Different Similarity Measures for Test Suite Optimization." *Advanced Science, Engineering and Medicine* 10.7-8 (2018): 833-836.
18. Singh, Shilpi, Chetna Sharma, and Udai Singh. "A simple technique to find diverse test cases." *International Journal of Computer Applications* 975 (2013): 8887.

AUTHORS PROFILE

Dr Shilpi Singh , Assistant Professor, Department of Computer Science and Engineering, ASET, Amity University Patna, Patna, India. She received her Ph.D. degree in Information Technology from Babasaheb Bhimrao Ambedkar University, Lucknow, UP, India. Her current research interests include software engineering, software testing and machine learning. She has published numerous papers in reputed national and international journals.

Dr Preetish Ranjan, Assistant Professor, Department of Computer Science and Engineering, ASET, Amity University Patna, Patna, India. He received her Ph.D. degree in information technology from Indian Institute of Information Technology, Allahabad, UP, India. His research area includes social network analysis, call data record analysis, detection of socio-technical attack. He has published numerous papers in reputed national and international journals.

Dr Raj Shree, Assistant Professor, Department of Information Technology, Babasaheb Bhimrao Ambedkar University, Lucknow, UP, India. India. She received her Ph.D. degree in information technology from Babasaheb Bhimrao Ambedkar University, Lucknow, UP, India. Her research area includes Image Processing, Software Engineering, Computer Communications (Networks) and Artificial Intelligence. She has published numerous papers in reputed national and international journals.