IJCSE International Journal of Computer Sciences and Engineering Open Access
Research Paper Vol.-6, Issue-7, July 2018 E-ISSN: 2347-2693

Test Case Prioritization Using Modified Bat Algorithm

S. Chaudhary[1], R. Singh[2]

Dept. C.S.E, Deenbandhu Chhotu Ram University of Science and Technology, Murthal, India

*Corresponding Author: rajvirsingh.cse@dcrustm.org , Tel.: 9599782183

Available online at: www.ijcseonline.org

Accepted: 12/Jul/2018, Published: 31/July/2018

***Abstract*—** In these research methods for Test case prioritization to perform schedule test case has been discussed. It has been discussed for running in an order that try to raise effectiveness. It has been discussed that goals is Average Percentage Fault Detection (APFD) . This paper, we describe several methods in case of prioritizing of test cases. Such work has been dependent on thought of prioritization of test case methods. It takes the objective to expose maximum fault potential. The mutant based testing is used on the primary basis to get the effectiveness of test cases. It is used to expose the known faults. It is based on the version information. It also considers the historical aspects of effectiveness. The effectiveness is tested against known faults like heuristics. The approach is a probabilistic estimate. It is made for test cases and mutants testing. The problem lies in finding the best sequence of the application. It is of test cases. The aim is to detect average percentage of APFD. This is over lifetime in case of test suite. It should be maximized. The outputs are suggesting that several methods could significantly improve. They are improved due to hybridization of algorithm execution time complexity of algorithm has been increased.

***Keywords*—** Test Case Prioritization, BAT, Levy Flight, APFD ,Metaheuristc Algorithms

I. INTRODUCTION

Prioritizing and Scheduling test cases have been most important work. They used in process of software testing. Here the priority is given to test cases. It is supporting in identifying what test case should be considered first. It is performed during performing component testing. The low priority test cases are left out. The prioritization of Test case is helpful in up gradation rate of error detection. In such cases testing becomes more faster for developers. This is almost not impossible to run each test case. It is must to check huge environment of software to be tested. It occurs at the time of testing phase of project. There should be test cases bases prioritized helps plan. This helps in fast and effective execution. It is performed at time of meeting deadlines. It is going to become more practical and achievable. Test case prioritization mechanism and procedure are needed have been discussed. There is requirement of prioritization methods due to following reasons:

1) Prioritization allow to choose the test cases to run first,
2) It is taking lot of time in order to execute the complete test suite.

Prioritization of Test case is helping in schedule of execution of test cases. It is done in a proper way to raise software testing efficiency. Huge amount of test cases has been made. They have been designed to test software products. It is impossible to run each test case. This is because of time limitation. Thus there has been requirement of prioritization of test cases during the testing process. Software Quality has been increased. The cost is decreased. Product would be delivered to customer timely [3], [4].

Background

TEST CASE PRIORITIZATION

The efficiency in case of software testing process has been increased by Test case prioritization. This has been compared to run of test cases. It has been done in non-prioritized order. The Software would be tested after running several test cases. These are tested in multiple circumstances. There would be redundancy if each test cases is run for common type of modules. It may be changed or unchanged. Prioritization of Test case would raise performance of testing. The process of checking the Effectiveness of Prioritization Techniques. During prioritization effectiveness [5],[6],[7] of prioritization techniques which are dependent on coverage have been compared. They have been compared using APFD metric.

Average Percentage in case of Fault Detection

It is calculated by Average Percentage in case of Fault Detection that how fastly. This has been possible to find out faults for the given test suites [6]. If the range of value is high, then it shows the faster rate of fault detection. 0 to 100 is Average Percentage Fault Detection values. By adopting APFD metric formula user would be able to compute how various components in case of software are effective. Here is a formula for APFD metric [6]:

$$APFD = 1 - \frac{TF1+TF2+\ldots\ldots+TFm}{nm} + \frac{1}{2n}$$

here, m has been considered as number of exposed faults n may be taken as total count of test cases TFi will be placed at first rank test case in T. It is exposing fault i.

II. RELATED WORK

Muhammed Maruf O¨ RK[8], Here an experiment was done. It was accomplished by using only six project data sets. It may be that this case misleads the validation of technique. BITCP is going to upgrade in fixing the values of loudness. It looks the challenges in the test cases. With this it initializes the population. Here the difficulties are habitual to loudness. It utilizes the maintainability of metrics. Besides, some appropriate metrics are needed in case conducting analysis. It should be in depth for the works of future.

Bo Jiang, Zhenyu Zhang[10], Tat Chee Avenue,T. H. Tse[9] It has been presented by group of adaptive random test case techniques for prioritization. It conducts an experiment for measuring its performance. It shows the ART prioritization methods. It sets the distance definitions. These definitions are explained at various levels of code coverage. Besides it utilizes to spread test cases. It is early as expected.
Serdar Carbas, [16], Here a study of bat-inspired optimization algorithm has been done. It is BIO. It has been defined as an effective method. The method is used for optimization of steel space frames. It is taken with variables of discrete sizing. Algorithms are employing general rules of technique that have been inspired by bat. Reformulation of techniques has carried out for application. It is for structural optimization for hardware and software system.

Suri, Bharti, [21], presents an implementation of an already introduced Ant Colony Optimization Algorithm for Test Case Selection and Prioritization. Graph representation and example runs explained in the paper show how the random nature of ACO helps to explore the possible paths and choose the optimal from them. Results show that ACO leads to solutions that are in close proximity with optimal solutions.

Yazan A. Alsariera, Mazlina A. Majid and Kamal Z. Zamli[13],[14], explain and conduct a real-world case study for t-way strategy based on meta-heuristic algorithm, namely Bat-inspired algorithm to benchmark Bat-Inspired t-way strategy for interaction testing. The case study results are encouraging, especially at the prospect of supporting a high interaction strength. The optimizing of the test suite of the case study is different among traditional strategies, which perform a simple number of processes during generation due to the use of the meta-heuristic algorithm.

Sahil Gupta et al. [17] Here has been proposed an approach for test case prioritization. It is done for improving the regression testing. Some analysis has made for some specific test cases which are prioritized and non-prioritized.

Shifa-e-Zehra et al. [18] Group of test case prioritization methods has been presented. It utilizes the dependency information. It is taken from a test suite. It is done in order to prioritize test cases. The nature of the techniques reserved the dependencies. This should be done in the test ordering. The APFD has been increased by these techniques. It is of fault detections. It has been achieved by untreated order. They may take random orders and test cases.

Praveen Ranjan Srivastava [19] It has been proposed an algorithm for prioritization of test case. It is done for improving regression testing. The prioritized and non-prioritized cases analysis has been made. Here the support of average percentage fault detection metric has been taken. Some graphs have proved that prioritized cases were more effective.

Kavitha,R.,[22], described regression test prioritization techniques reorder the execution of a test suit in an attempt to ensure that faults are revealed at the earlier stage of the testing process. Test case prioritization techniques schedule test cases for execution so that those with higher priority, according to some criterion are executed earlier than those with lower priority to meet some performance goal. In this paper an algorithm is proposed to prioritize test cases based on rate of fault detection and fault impact. The proposed algorithm identifies the severe fault at earlier stage of the testing process and the effectiveness of prioritized test case and comparison of it with unprioritized ones with the help of APFD.

III. METHODOLOGY

Process Flow of proposed work
1. The input to the proposed work is as follow:
FAULT MATRIX
Fault matrix represents the 0 and 1 row and column wise. Here 0 represents the false. The 1 represents true. When such data is organized in two dimensional form then it is know as fault matrix. This would be input to the proposed model. APFD , BAT Algorithm and random walk algorithm would process this input.

2. The objective functional parts of proposed work are as follow:
APFD (Average Percentage Faults Detected)
Test case prioritization techniques are consisted of scheduling test cases. This is an order. It is the improving performance in the case of regression testing. It has been considered inefficient. There they have to re execute every test case. It should be in case of each program function. Also it is in occurrence of modification. The test case prioritization methods are arranged in test cases in the test suite. It orders in the most beneficial order. So it allows for the increment in testing usefulness. Here the performance objective is i.e. fault detection rate. It is a medium how quickly faults are detected at the testing process time.
BAT ALGORITHM

The Bat algorithm is the algorithm which is metaheuristic. It may be considered in case of global optimization. The motivation was the echolocation behaviour of the microbars. This has been the diversing pulse rates of secretion and loudness. The praise of echolocation of tells us that each virtual bat flies arbitrarily. It flies with velocity. It's position is the changing wavelength. The loudness is also changed. It changes frequency and loudness. This also diversifies pulse emission rate.. It is in the search of its prey. The choice of the best goes on until meeting the fixed criteria . This uses a method which is frequency-tuned. It is for controlling varying behaviour of the swarm of bats. It also makes balance between exploration and exploitation. In bat algorithm, it is done by tuning algorithm-dependent parametersRANDOM WALK ALGORITHM

The random walk is a mathematical object. This may be also defined] as a stochastic process. It is consists of a sequence of random steps. Here in this case few mathematical spaces are taken as integers. A basic example of a random walk has taken as a random walk. It is in the case of integer number line, z. It starts with 0. At each step it moves +1 or −1 with equal probability. There are other many examples. These are consisted of path traced by a molecule as it travels in a liquid or a gas. We should consider search path of a pseudo animal. Random walks have been basic topic while explaining of Markov processes. Here the mathematical study is extensive. There are various properties. These are consists of some dispersal distributions. There is a similarity with first-passage or hitting times. It includes encounter rates or transience. These are introduced for quantifying their behaviour.

OPTIMIZED PRIORITY SEQUENCE

A particular order or sequence in which things takes place. A priority is based on a predetermined assignment of value. It is also based on importance, to different types of events and people. This module is to optimize priority of sequence.

The OUTPUT of the proposed work is as follows

APFD is the Average of Percentage Faults detected. The effectiveness of algorithm is illustrated with help of presented results by APFD metric. Here it aims at determining the effectiveness of prioritized and non-prioritized case. For this it takes the help of APFD.

TIME COMPLEXITY

Time complexity is a computational complexity. It explains amount of time. It takes time to run an algorithm. Complexity of Time is normally considered by calculating number of basic operations. These may be done by algorithm. It supposes a preset amount of time for performance of each basic operation. Algorithm do the number of basic operations. They are differentiated by the most constant factor. Here the running time of an algorithm's can change between different inputs. These inputs may have same size. Worst-case time complexity may be taken into consideration commonly. This is maximum amount of time required for given sized inputs. Here we take the average of time on inputs of the given size. In both cases, Time complexity is shown as a function of size of input. It is very complex to calculate this function accurately. The running time for the small inputs is not consequential; The behavior of complexity focused commonly. It is when input size increases. There is asymptotic behavior of complexity

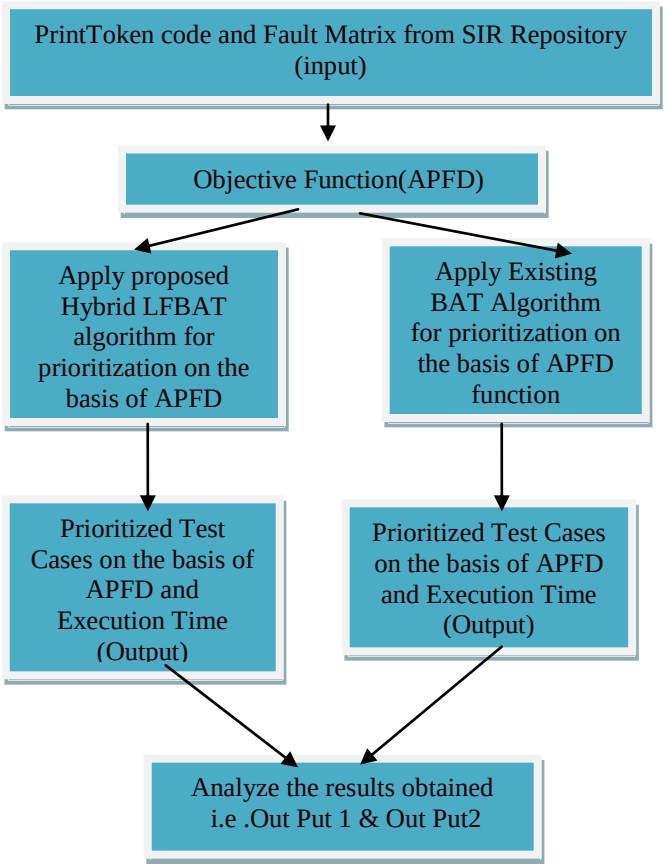

fig 1: Proposed work model

The most significant tasks in the software testing process is Prioritizing test cases. Test cases are prioritized. It helps to find out which test cases should be kept on high priority. It also considers while performing priority testing which test cases should be kept on low priority. It should see what cases can be left out. It helps to improve Average Percentage Fault Detection in testing.

IV. RESULTS AND DISCUSSION

The questions of our research have been answered by conducting extensive experiments. The results were collected. These were analyzed statically. These run for many times on three program versions. These are used for

overcoming the internal pressure to validity. Every algorithm was executed for50 iteration. It is used in a single run. Table 1 below shows details of the experimental data sets.

Table 1. Experimental Data Sets.

Program Version	Total no. of test cases	Total no. of faults
Flex V1	567	19
Printtoken	200	200
Scheduling	200	200

The following table is representation of number selected APFD and time complexity in case of existing BAT and proposed –LF. From following table it is found that the time complexity of proposed –LF is more as compare to exiting –BAT. Shows in table 2.

Table 2. Comparison of Existing and Proposed algorithms.

	Existing-BAT			Proposed-LF	
Num Selected	APFD	Time Complexity	Num Selected	APFD	Time Complexity
200	99.75	6.8361	200	99.75	7.3261
200	99.75	6.9109	200	99.75	7.5347
200	99.75	6.8513	200	99.75	7.5797
200	99.75	6.8776	200	99.75	7.6366
200	99.75	6.9584	200	99.75	7.6444
200	99.75	6.8745	200	99.75	7.6249

The following table is representation of Iteration, Base APFD and time complexity in case of base APFD and proposed APFD. From following table it is found that the time complexity of proposed APFD is more as compare to BASE APFD.Table 3 shows the collected matrix at different iteration.

Table 3: Collected Matrix at different Iteration

Iteration	Base APFD	Time Complexity	Proposed APFD	Time Complexity
10	99.75	1.7531	99.75	1.9558
20	99.75	3.0855	99.75	3.4419
30	99.75	4.2480	99.75	4.7707
40	99.75	5.6394	99.75	6.2108
50	99.75	6.8569	99.75	7.5369

The following table is representation of dataset, BAT APFD, BAT time complexity, LFBAT proposed APFD, LFBAT proposed time complexity. From following table it is found that the time complexity of proposed LFBAT is more as compare to BAT APFD. Table 4 shows the Some Best value of Matrics Used with datasets in existing and proposed algorithms BAT, LFBAT (APFD , Time Complexity).

Table 4: Some Best value of Matrices BAT, LFBAT (APFD , Time complexity).

Dataset	BAT-APFD	BAT-TimeComplexity(sec)	LFBAT-proposed APFD	LFBAT-Proposed Time Complexity(sec)
Flex V1	99.85	2.916	99.87	4.8332
Printtokens	99.83	6.7182	99.87	8.3974
Schedule	99.83	6.7104	99.86	8.7702

The comparison chart in case of BAT TIME COMPLEXITY AND LFBAT TIME COMPLEXITY is as follow:

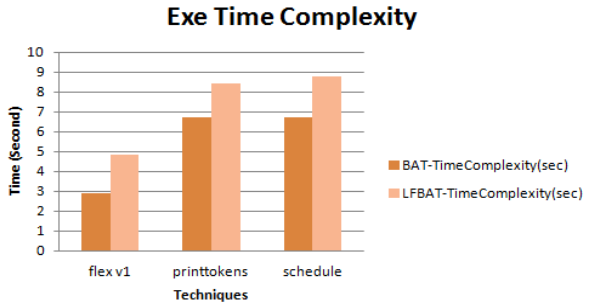

Fig 4: BAT, LFBAT time complexity comparing graph

The comparison chart in case of BAT , LFBAT, APFD is as follow:

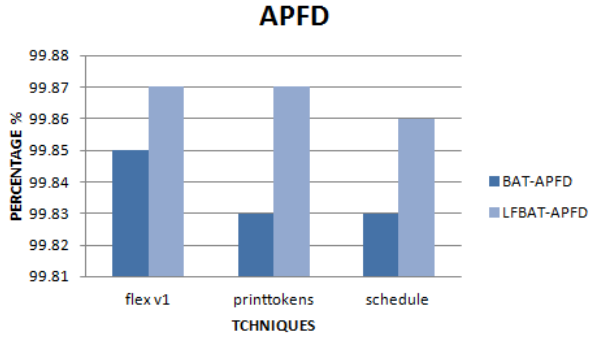

Fig 5: BAT, LFBAT APFD comparing graph

MATLAB 7.0 is used in implementation of bat algorithm, and in comparison of both APFD and Time Complexity, APFDLFBAT and Time Complexity LFBAT. It is a software tool which is high performance language for technical computing. This software is used here to calculate gbest value in bat and hybrid leavy flight.

V. CONCLUSION and Future Scope

This research is based on improved test case prioritization method. These are Levy Flight. It has been proposed method. The method has been tested on three software projects namely Flex V1, Printtoken , Scheduling and we measure two performance out APFD and Time Complexity based on both of algorithms Existing and Proposed. As a result ,LFBAT is superior to BAT and LF with respect to APFD result. The time complexity is reduced and APFD value is increased. The proposed algorithm performed better in both criteria (APFD

,Time complexity). In future we can use other new metaherustic techniques for further improvement.

REFERENCES

[1] B. Korel and J. Laski, "Algorithmic Software fault localization", International Conference on System Sciences, pp. 246-252, 1991.

[2] Dennis Jeffrey and Neelam Gupta, "Test Case Prioritization Using Relevant Slices", Proceedings of the 30th Annual International Computer Software and Applications Conference, Vol. 1, pp. 411-420, 2006.

[3] J. Karlsson and K. Rayan, "A Cost Value Approach for Prioritizing Requirements", IEEE Software, Vol. 14, No. 5, 1997.

[4] Alexy G. Malishevsy, Gregg Rothermel and Sebastian Elbaum, "Modeling the Cost- Benifts Tradeoffs for Regression Testing Techniques", Proceeding of the International Conference on Software Maintenance (ICSM), 2002.

[5] Malishevsky A.G., J.R. Ruthruff, G. Rothermel and S. Elbaum, "Cost-cognizant test case prioritization technical Report", TR-UNL-CSE-2006-0004, 2006.

[6] H. Do and G. Rothermel, "On the use of Mutation faults in Empirical Assessments of Test case prioritization Techniques", IEEE Transaction on Software Engineering., Vol. 32, No. 9, 2006.

[7] Zheng Li, Mark Harman and Robert M. Hierons, "Search Algorithms for Regression Test Case Prioritization", IEEE Transaction on Software Engineering, Vol. 33, No. 4, pp. 225-237, 2007.

[8] Lionel C. Briand, Victor R. Basili, Christopher J. Hetmanski, "Developing Interpretable Models with Optimized Set Reduction for Identifying High-Risk Software Components", IEEE Transactions on Software Engineering, Vol. 19, No. 11, pp. 1028-1044, 1993.

[9] Garlan, C., R. Allen and J. Ockerbloom, "Architectural Mismatch or Why It's Hard to Build Systems Out of Existing Parts", Proceeding of 17th International Conference on Software Engineering, pp. 179–185, 1995.

[10] L. Mei, W. K. Chan, T. H. Tse, and R. G. Merkel, "Tag-Based Techniques for Black-Box Test Case Prioritization for Service Testing," in 2009 Ninth International Conference on Quality Software, 2009, pp. 2130.

[11] Elbaum S.,G. Rothermel, S. Kanduri and A.G. Malishevsky, "Selecting a cost-effective test case prioritization Techniques", Software Quality Journal, Vol. 12, pp. 185-210, 2004.

[12] Malishevsky A.G., J.R. Ruthruff, G. Rothermel and S. Elbaum, "Cost-cognizant test case prioritization technical Report", TR-UNL-CSE-2006-0004, 2006.

[13] H. Do and G. Rothermel, "On the use of Mutation faults in Empirical Assessments of Test case prioritization Techniques", IEEE Transaction on Software Engineering., Vol. 32, No. 9, 2006.

[14] Zheng Li, Mark Harman and Robert M. Hierons, "Search Algorithms for Regression Test Case Prioritization", IEEE Transaction on Software Engineering, Vol. 33, No. 4, pp. 225-237, 2007.

[15] A. Panichella, R. Oliveto, M. Di Penta, and A. De Lucia, "Improving Multi-Objective Test Case Selection by Injecting Diversity in Genetic Algorithms," IEEE Trans. Softw. Eng., vol. 41, no. 4, pp. 358383, Apr. 2015.

[16] Jeffery A. Solheim and John H. Rowland, "An Empirical Study of Testing and Integration Strategies using Artificial Software Systems", IEEE Transactions on Software Engineering, Vol. 19, No. 10, pp. 941-949 , 1993.

[17] Sahil Gupta, Himanshi Raperia, Eshan Kapur, Harshpreet Singh and Aseem Kumar, "A Novel Approach for Test Case Prioritization" International Journal of Computer Science, Engineering and Applications (IJCSEA) Vol.2, No.3, pp. 53-60, 2012.

[18] Shifa-e-Zehra Haidry and Tim Miller, "Using Dependency Structures for Prioritization of Functional Test Suites", IEEE Transactions on Software Engineering, Vol. 39, No. 2, 2013.

[19] Praveen Ranjan Srivastava, "Test Case Prioritization", Journal of Theoretical and Applied Information Technology, pp. 178-181, 2008.

[20] Rothermel G, Elbaum, S, Kinneer, A and Do H, 2006. Software-artifact infrastructure repository. URL http:// sir.unl. edu/portal.

[21] Suri, Bharti, and Shweta Singhal. "Implementing ant colony optimization for test case selection and prioritization." *International journal on computer science and engineering* 3.5 (2011): 1924-1932.

[22] Kavitha, R., and N. Sureshkumar. "Test case prioritization for regression testing based on severity of fault." *International Journal on Computer Science and Engineering* 2.5 (2010): 1462-1466.

Authors Profile

Rajvir Singh is an Assistant Professor in Computer Science & Engineering Department , Deenbandhu Chhotu Ram University of Science & Technology (DCRUST) (A State Govt. University), Murthal: 131039 (India) since 5th Feb., 2010 to till date. Before joining DCRUST he has served in Delhi College of Engineering / Delhi Technological University, Delhi: 110042 (India) from 19th July, 2007 to 5th Feb., 2010 as Contractual Lecturer (Computer Engineering). He has completed his B.E.(CSE) degree in 2003 from MDU, Rohtak, Haryana:124001(India) and M.Tech. in Computer Engineering in 2007 from YMCA Institute of Engineering / YMCA University of Science & Technology, Faridabad, Haryana, India. He is pursuing Ph.D.(CSE)(Part-Time) from DCRUST, Murthal:131039(India). He have qualified various national level tests like GATE-2003, UGC-NET-2009(June), UGC-NET-2012(June), Research Aptitude Test (RAT)-2010 conducted by Jamia Millia Islamia, New Delhi-110025(India). His areas of interest are Software Testing, Software Engineering, Data Base Management Systems, Computer Graphics, Theory of Automata Computation and Artificial Intelligence.

Shivani Chaudhary is pursuing M.tech (Computer Science & Engineering) from Deenbandhu Chhotu Ram University of Science & Technology (DCRUST), murthal. She has completed her B.tech (Computer Science & Engineering) from SITM, Sonipat, India in 2016

