Profiling of Test Cases with Clustering Methodology

Fayaz Ahmad Khan
Department Of Computer Science and Applications, B.U. Bhopal (M.P.)

Anil Kumar Gupta, Ph.D.
Department Of Computer Science and Applications, B.U. Bhopal (M.P.)

Dibya Jyoti Bora
Department Of Computer Science and Applications, B.U. Bhopal (M.P.)

ABSTRACT
Software testing is an imperative task in software development process. Software testing is used to identify the correctness, completeness and quality of the software product or system. Till date, software testing is considered as a very expensive activity as it takes a lot of testing efforts, time and cost to perform it. One of the expansive factors behind is the design or generation of effective test cases for a particular software product. In this paper, we are trying to find out the effective test cases from the generated whole set on the basis of clustering methodology so that the size of test suit is reduced and redundant test cases are eliminated automatically. Here, we are following the famous K-Means algorithm with a proper distance measure.

Keywords:
Software Engineering, Software Testing, Test Cases, Clustering, K-Means

1. INTRODUCTION
Testing software as known is very important and challenging activity. Lack of testing has resulted in many software related troubles in past, and have actually brought social problems and financial losses. Software testing is an effective method for estimating the present reliability and predicting future reliability of the software. We test software by selecting appropriate testing technique and applying them systematically. Testing techniques refer to different methods of testing particular features of a computer program, system or product [1]. We have to make sure that we select technique(s) that will help to ensure the most efficient and effective testing of the system [2]. Test techniques should find the greatest possible number of errors with manageable amount of efforts applied over a realistic time span with a finite number of test cases.

Test case generation is an imperative activity in the software development process. There are usually two approaches for software testing: white box testing and black box testing. In white box testing approach the code or internal structure of a program is analyzed by the application of many white box test case generation techniques like statement coverage, decision coverage, condition coverage and combination of these techniques. In case of black box technique, which is also called data driven, or input / output driven, views a program as a black box and thus is uncertain about the internal behavior and structure of the program. The techniques used for test case generation are Equivalence partitioning, boundary value analysis, cause- effect graphing and error guessing. In literature many automatic test case generation techniques and tools have also been proposed and developed which some extent reduce the testing effort, time and cost. But there is still a long way to go as results are very inconclusive.

To evaluate the goal of software testing activity, test objectives are usually defined before the testing process starts [11]. These objectives may vary from each other and will be of different granularity. Thus a single test case from an infinite input domain can hardly satisfy all the test requirements. To find an optimal or sub set of test cases from usually large infinite domain of test cases that can satisfy the same requirements as the original test suite is a research problem known as test suite reduction or test suite minimization.

In literature many efforts have been put into research on how to generate and reduce the size of test suite while maintaining its effectiveness include heuristic approach[3,4,5,6,7], the genetic algorithm based approach[8], IPL based [9,10]. But in recent literature many new approaches have been also proposed. Saraph et al [12] proposed automatic Test case generation and reduction technique in which employment application approval system is used as prototype and for pruning and feature selection neural network is used for test case generation. In [13], L. Ramesh and Uma used state charts for test case generation and then applied clustering for the reduction of test cases. Also in [14], Jevarathinam and Thanamani, use symbolic execution based frame work for the generation of test cases. In [15], Trudso Egholm reported an efficient approach in which they compare equivalent partitioning, boundary value analysis and pex white box technique and proposed a methodology for black and white box testing. Muthyala and Naido in [16], presented a novel test suite reduction technique using clustering approach.

2. CLUSTERING METHODOLOGY
Data clustering is a very essential area of data mining [17]. Data clustering is an unsupervised machine learning technique which allows making natural groups of data to determine the patterns from the data. The goal is that the objects in a group will be similar (or correlated) to one another and different from (or dissimilar) with the objects in other groups. We can also say that in clustering the objects in a cluster posses low inter-class similarity within the cluster and high intra-class similarity outside the cluster [18]. Clustering algorithms are primarily divided into two methods: hierarchical and partitional clustering. In hierarchical methods, the dataset of n objects is decomposed into a hierarchy of groups and the result is represented by a tree structure diagram known as dendrogram, whose root node represents the whole dataset and each leaf node is a single object of the dataset. There are usually two general approaches for the hierarchical clustering: agglomerative and divisive [17] [18]. Agglomerative approach is a bottom up approach starting with n-leaf nodes as individual clusters, moving upwards towards the root with some merging criteria. While divisive hierarchical clustering technique is a top down approach, which initially treats the whole data set as a single cluster or root node and gradually

splits data into different clusters downwards based on the certain properties of the data. While in case of partitional clustering, the datasets of n objects is divided into a pre-defined number (k) of clusters, where k<= n. There are various partitional clustering algorithms of which, K-Means is a very popular one because of its low complexity [18]. K-Means algorithm works as follows: Given k, the k-means algorithm [17][19] is implemented in 4 steps:

Partition objects into k nonempty subsets.

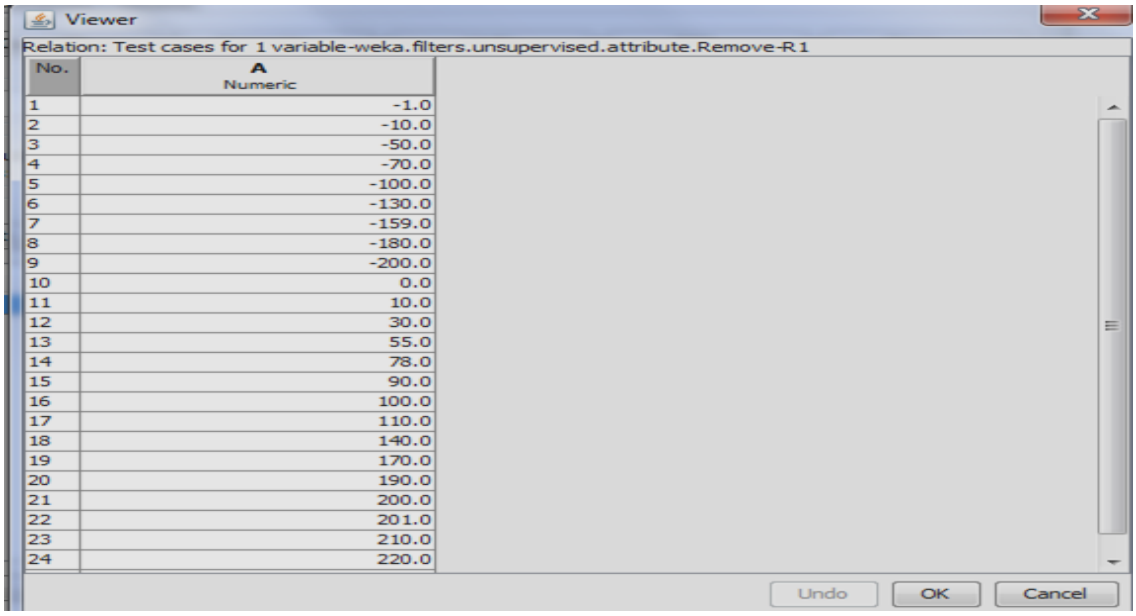
Compute seed points as the centroids of the clusters of the current partition. The centroid is the center (mean point) of the cluster.

Assign each object to the cluster with the nearest seed point.

Repeat steps 2 and 3 until the centroids don't change.

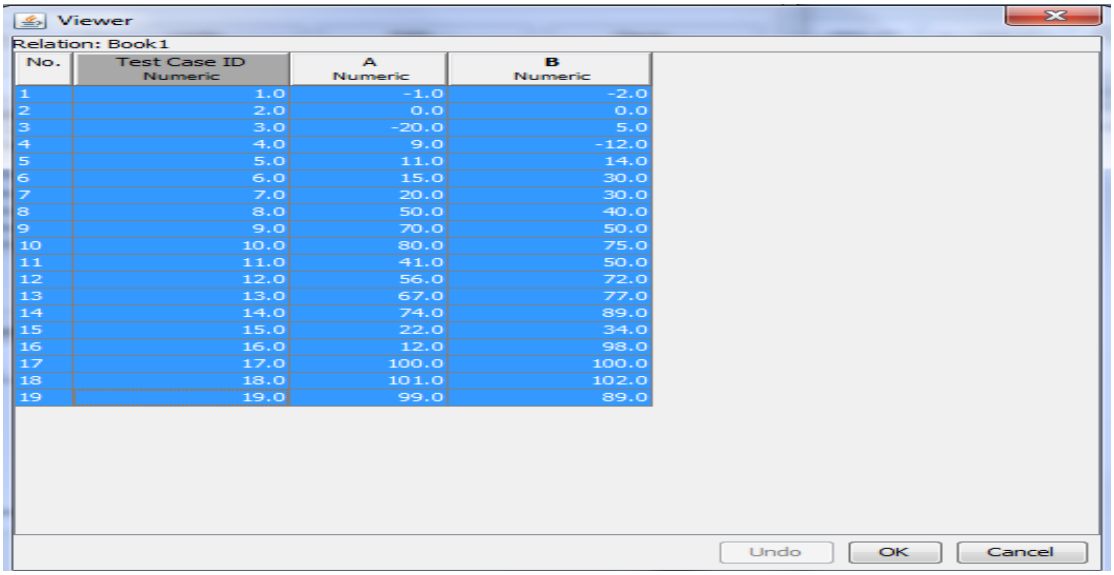
3. EXPERIMENT

In this paper we have applied K- Means approach on two test suites: one test suite in Fig. 1 is generated for a single variable input case and second test suite in Fig. 2 for two variables input case using two very well known black box test case generation techniques (equivalence partitioning, boundary value analysis with Robustness testing). We included all the possible values including out of range and negative ones in addition to test cases generated by using Equivalence partitioning and Boundary value analysis. As it is evident from both the Fig. 1 and Fig. 2, that the generated test cases are redundant and large in number, which would result in wastage of effort, money and time. So to automatically minimize the test cases in the test suite, we try to get a representative set of test cases from the whole test cases (that will be effective and non- redundant); K-Means algorithm is applied on both the test suites (in Fig. 1 & Fig.2. The result of experiment is shown in Fig.3 and Fig 6. The Fig.5 represents the whole structure and implementation of proposed work. The visual automatic cluster assignment for both cases is also shown in Fig. 4 and Fig. 7.



No.	A Numeric
1	-1.0
2	-10.0
3	-50.0
4	-70.0
5	-100.0
6	-130.0
7	-159.0
8	-180.0
9	-200.0
10	0.0
11	10.0
12	30.0
13	55.0
14	78.0
15	90.0
16	100.0
17	110.0
18	140.0
19	170.0
20	190.0
21	200.0
22	201.0
23	210.0
24	220.0

Fig. 1: Test Cases for one input variable.



No.	Test Case ID Numeric	A Numeric	B Numeric
1	1.0	-1.0	-2.0
2	2.0	0.0	0.0
3	3.0	-20.0	5.0
4	4.0	9.0	-12.0
5	5.0	11.0	14.0
6	6.0	15.0	30.0
7	7.0	20.0	30.0
8	8.0	50.0	40.0
9	9.0	70.0	50.0
10	10.0	80.0	75.0
11	11.0	41.0	50.0
12	12.0	56.0	72.0
13	13.0	67.0	77.0
14	14.0	74.0	89.0
15	15.0	22.0	34.0
16	16.0	12.0	98.0
17	17.0	100.0	100.0
18	18.0	101.0	102.0
19	19.0	99.0	89.0

Fig. 2: Test cases for two input variables.

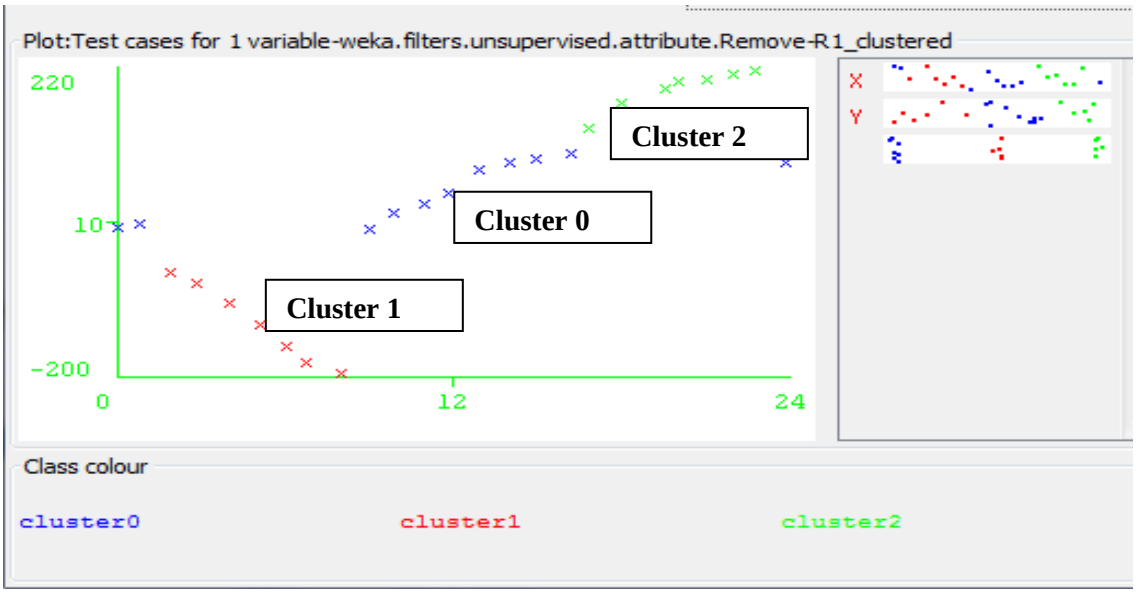


Fig. 3: Cluster Assignment and visualization.

Test case No.	Test Data	Result
0,	-1,	cluster0 }
1,	-10,	cluster0 } Overlap
2,	-50,	cluster1
3,	-70	cluster1
4,	-100,	cluster1
5,	-130,	cluster1
6,	-159,	cluster1
7,	-180,	cluster1
8,	-200,	cluster1
9,	0,	cluster0
10,	10,	cluster0
11,	30,	cluster0
12,	55,	cluster0
13,	78,	cluster0
14,	90,	cluster0
15,	100,	cluster0
16,	110,	cluster0
17,	140,	cluster2
18,	170,	cluster2
19,	190,	cluster2
20,	200,	cluster2
21,	201,	cluster2
22,	210,	cluster2
23,	220,	cluster2
24,	90,	cluster0 → Overlap

Fig. 4: Result generated after implementation of K-Means Algorithm for one input Variable.

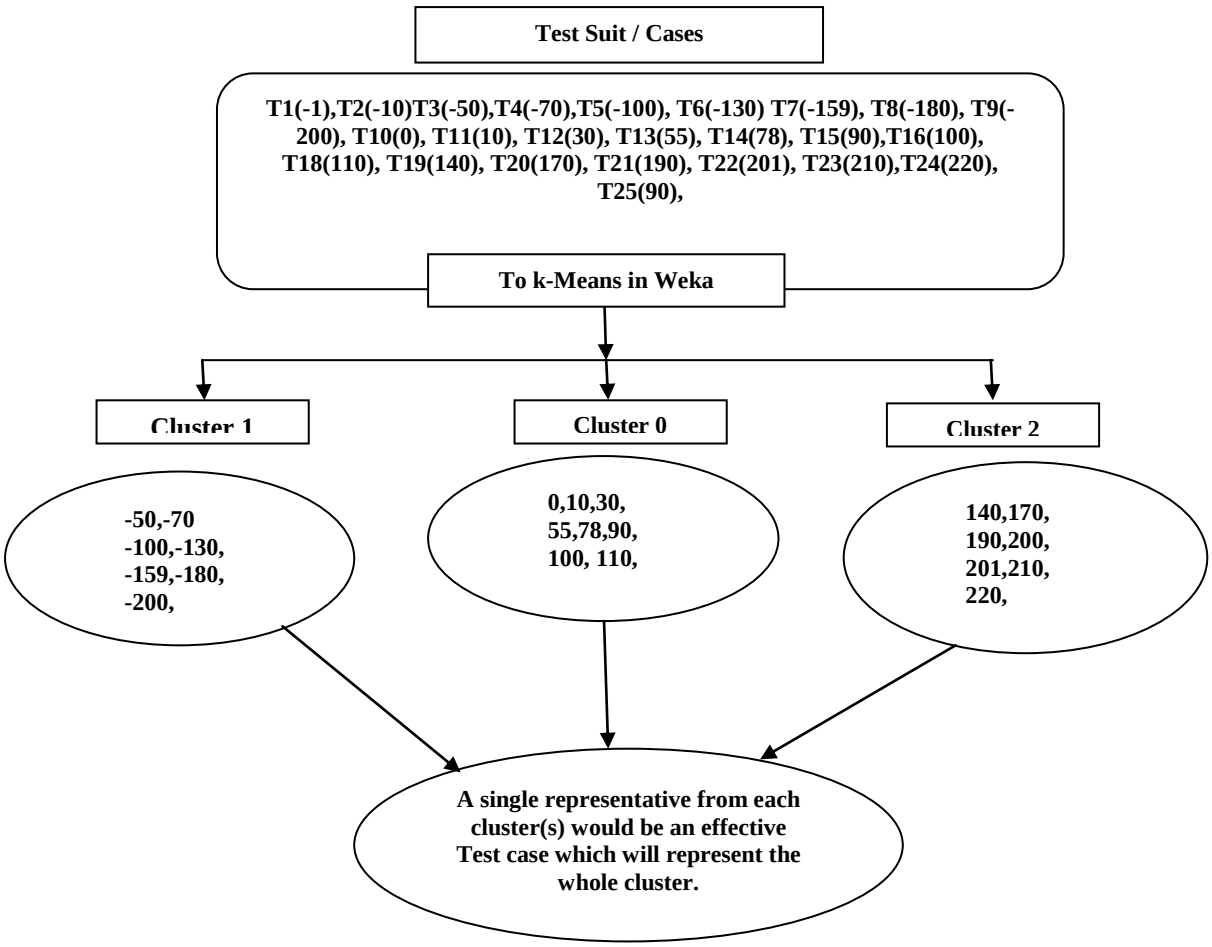


Fig. 5: The proposed approach

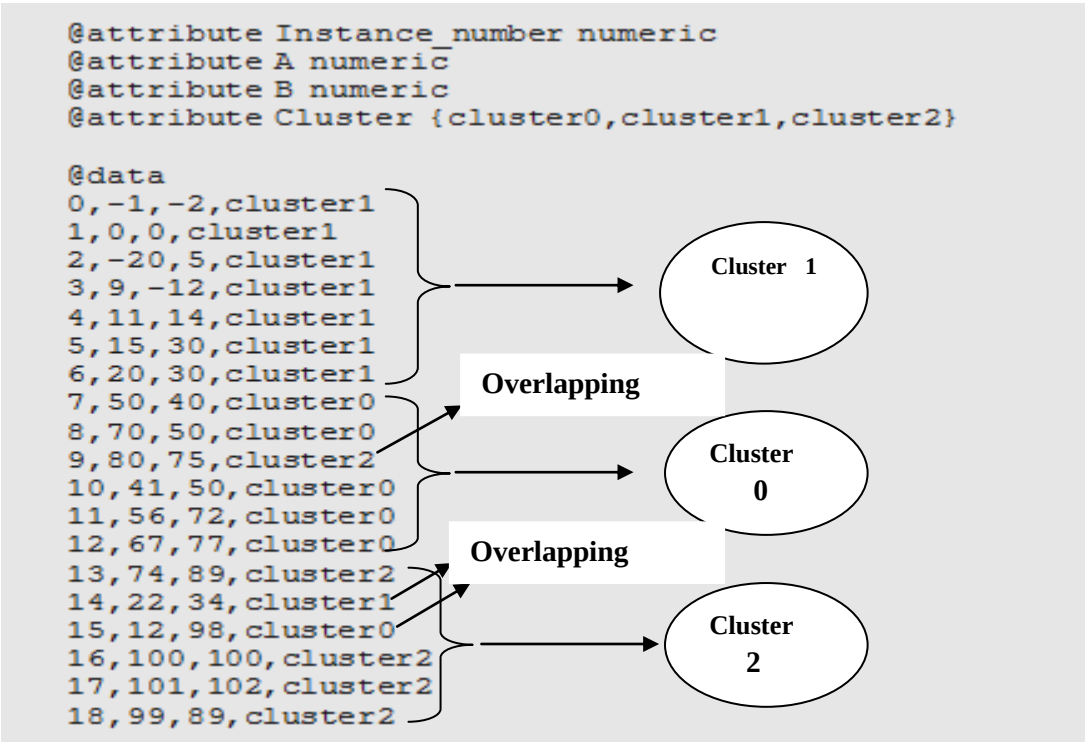


Fig. 6: Result generated after implementation of K-Means Algorithm for two input Variables.

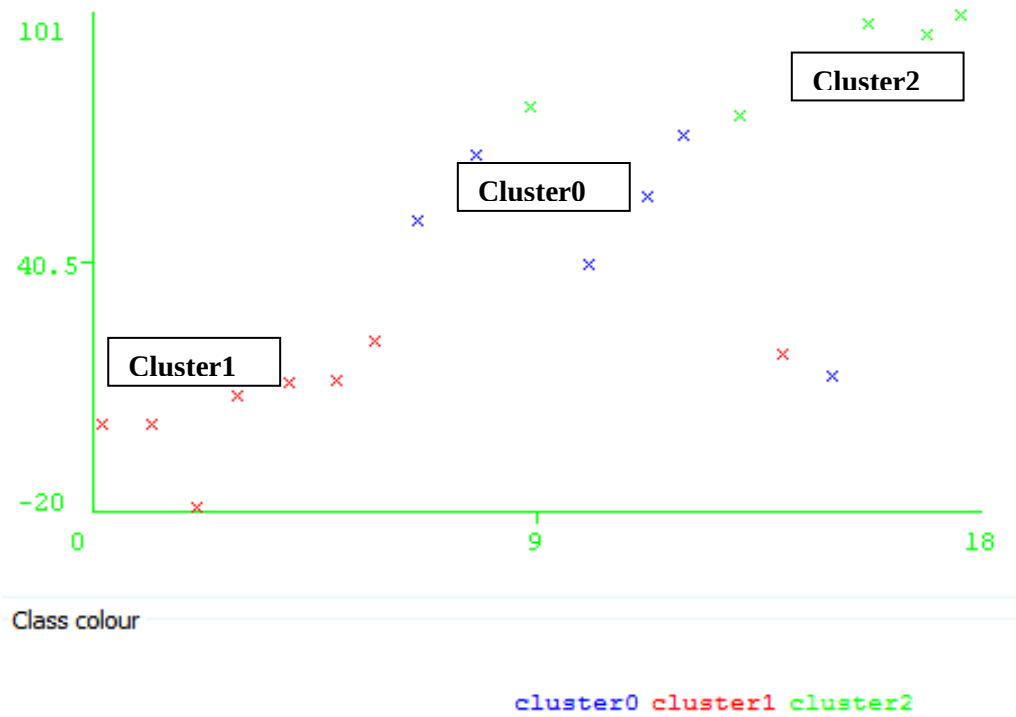


Fig. 7: Visual Cluster Assignment in Weka.

4. CONCLUSION

With the application of K-Means hard clustering approach, we have partitioned the both test suites into a pre specified number (k=3) of profiles or clusters as Cluster (0), Cluster (1) and Cluster (2), using most commonly used proximity measure known as "Euclidean" distance measure. Hence, instead of testing with the whole set of test cases, we have formed a profile or cluster so that a single representative from each profile will be an effective and an efficient alternative. Thus, we have reduced both the size and redundancy of a test suite automatically by profiling or clustering the same. The proposed approach will also decrease the testing effort, time and cost due to the automatic generation of an effective and non redundant test cases. Our current proposed approach is best suitable for test cases which are exclusive and exhaustive in nature. But due to the fuzzy nature of test cases, overlapping of certain test cases is observed and our future work will be to deal with such overlapping test cases with more appropriate method.

5. REFRENCES

[1] S. M. K. Quadri and Sheikh Umar Farooq, "Testing Techniques Selection: A Systematic Approach", Proceedings of the 5th National Conference; INDIACom-2011, pp-279-281, March 10 – 11, 201.

[2] Sheikh Umar Farooq, and SMK Quadri, "Identifying some problems with selection of software testing techniques", Oriental Journal of Computer Science & Technology Vol. 3(2), 266-269 (2010.)

[3] M.J. Harrold, R. Gupta, M.L. Soffa," A methodology for controlling the size of a test Suite", ACM Transactions on Software Engineering and Methodology 2 (3) (1993) 270–285.

[4] T.Y. Chen, M. Lau, "Dividing strategies for the optimization of a test suite", Information Processing Letters 60 (3) (1996) 135–141.

[5] J. Offutt, J. Pan, J. Voas, "Procedures for reducing the size of coverage based test sets", in:Proceedings of 12th International Conference on Testing Computer Software, 1995, pp. 111–123.

[6] J. Horgan, S. London, ATAC: "A data flow coverage testing tool for C", In: Proc. Symposium of Quality Software Development Tools, 1992,pp. 2–10.

[7] T.Y. Chen, M. Lau, "A new heuristic for test suite reduction", Information and Software Technology 40 (5-6) (1998) 347–354.

[8] N. Mansour, K. El-Fakih, "Simulated annealing and genetic algorithms for optimal regression testing", Journal of Software Maintenance 11 (1) (1999) 19–34.

[9] J. Hartmann, D. Robson, "Revalidation during the software maintenance Phase", Proceedings of International Conference on SoftwareMaintenance (1989) 0–80.

[10] J. Black, E. Melachrinoudis, D. Kaeli, "Bi-criteria models for all-uses test suite reduction", in: Proceedings of 26th International Conference on Software Engineering, IEEE ComputerSociety, Washington,DC, USA, 2004, pp. 106–115.

[11] G. Myers, "Software Reliability Principles and Practice", John Wiley and Sons, New York, 1979.

[12] P. Saraph, M. Last and A. Kandel, "Test case generation and reduction by automated input- output analysis", vol. 1, pp. 768-773.

[13] L. Raamesh and G. V. Uma, "Reliable Mining of Automatically Generated Test Cases from Software Requirements Specification (SRS)", International Journal of Computer Science, vol. 7, no. 1, (2010), pp. 87-91.

[14] R. Jeevarathinam and A. S. Thanamani, "Towards test cases generation from software specifications,"

International Journal of Engineering Science and Technology 01/2010, vol. 2, no. 11, (2010), pp. 6578-6584.

[15] S. Trudsø and K. Egholm, "Efficient test case generation Reliable Software and Architecture Project", (2010).

[16] K. Muthyala and R. N. P, "A novel approach to test suite reduction using data mining", Indian Journal of Computer Science and Engineering, vol. 2, no. 3, (2011), pp. 500-505.

[17] K. Jain, M. N. Murty and P. J. Flynn, "Data Clustering: A review", ACM Computing Surveys, vol. 31, no. 3, 1999.

[18] Dibya Jyoti Bora, Anil Kumar Gupta," A Comparative study Between Fuzzy Clustering Algorithm and Hard Clustering Algorithm", International Journal of Computer Trends and Technology (IJCTT) ,volume 10 number 2 – Apr 2014,pp. 108-113

[19] Dibya Jyoti Bora, Anil Kumar Gupta, "Effect of Different Distance Measures on the Performance of K-Means Algorithm: An Experimental Study in Matlab", IJCSIT) International Journal of Computer Science and Information Technologies, Vol. 5 (2) , 2014, pp. 2501-2506