Improving Partitioning Of ARPT By Using Clustering Technique

K. Devika Rani Dhivya, Dr.V.S. Meenakshi

Abstract: Software testing is normally accepted technique for assessing software quality. Whole software testing comprises successively running all the tests in the test suite. This is exhaustive and will take an inordinate volume of time. However, gathering and running a large test suite or test cases is still infeasible, as it would not be likely to run all test cases. An approach that grouping of Adaptive Testing and Random Partition Testing which is called as Adaptive Random Partition Testing (ARPT) strategy that enhances the testing ability. The objective of this proposed work is to improve random partition in ARPT strategies by utilizing clustering algorithms like Expectation Maximization (EM) algorithm and Non-negative Matrix Factorization (NMF) clustering algorithms and the Self-organizing map (SOM) which can be efficiently utilized for partition. In this way, random partitioning is improved to reduce the time conception and complexity in ARPT testing strategies. Improved ARPT (IARPT) utilizes clustering algorithms to improve the random partitioning of ARPT testing strategies. This paper describes how the clustering algorithms reduce the computational complexity of random partitioning testing in the ARPT testing strategy.

Index Terms: Software testing, Adaptive Testing, Random Partition Testing, Adaptive Random Partition Testing, Expectation Maximization algorithm, Non-negative Matrix Factorization, K- means,Self-organizing Map

1 INTRODUCTION

Adaptive Random Partition Testing (ARPT) [12] is a software testing procedure that utilized adaptive testing and random partition testing in an alternate manner. There are two options that are available to combine the AT and RPT in a testing process. One of the options use AT to pick a certain number of test cases and afterward use RPT to select a number of test cases before returning to AT. This strategy is called as ARPT-1 testing strategy. The other options use AT to choose the first n test cases and then switch to RPT for the remaining tests. This strategy is called as ARPT-2 testing strategy. However, the computation complexity is high for the random partitioning of ARPT. Hence in this research work, a different clustering algorithm is introduced to improve the random partitioning in ARPT strategies. The different clustering algorithms such as Expectation-Maximization (EM), K-means, Non-Negative Matrix Factorization (NMF) and Self-Organizing Map (SOM) can be effectively utilized for partition. EM algorithm partitions the test cases by maximizing the objective function which measures the goodness of the partitioning. In the K-means algorithm parting the test cases by joining comparative kind of test cases depending on the cosine similarity measure. Non-Negative Matrix approximation is the base of NMF for clustering the test cases. The winner unit which is determined based dependent on the Euclidean distance method was utilized in the SOM algorithm for clustering the test cases. Based on the winner unit choice, the weights are reorganized for that particular winner unit competitive learning rule. Thus the proposed ARPT with the clustering algorithm solves the parameter space of the problem between the target method and objective function of the test data.

- *K. Devika Rani Dhivya,*, Research Scholar, Bharathiar University, Coimbatore, Tamilnadu, India, devika58@gmail.com.*
- *Dr. V. S. Meenakshi, Assistant Professor, Department of Computer Science, Chikkanna Government Arts College, Tirupur, Tamilnadu, India, meenasri70@yahoo.com*

It achieves high test case coverage within less time and produces better accuracy. Finally, the performance of the ARPT with the Clustering algorithm is better and defect detection is high and less time to coverage of all test cases rather than the other methods.

2 LITERATURE SURVEY

This study focuses on Random Testing (RT), Partition Testing (PT), *Adaptive testing (AT) and various clustering methods for improving partitioning of test cases in software testing. During the course of the past few years, testing had grownup immensely with their techniques. Lots of new methodologies are developed which progress the softwaretesting performance by considering time, defect detection, and code coverage. Among them, some methods express their hard presence in particular areas and are related to their work are taken here as literature.

Table 1
Comparative study

S.No	Author Name & Year	Clustering algorithm	Time Taken	Fault Detection	Code / Test Case Coverage
1.	Sapna, P. G., & Mohanty, H. 2010[18]	Agglomerative Hierarchical Clustering (AHC) technique	Reduced	Improved	Improved
2.	Upadhyay, A. K., & Misra, A. K. 2012[12]	Average Percentage of Faults Detected technique. K-means clustering	Reduced	Improved	-
3.	Upadhyay, A. K. & Misra, A. K. 2012 [13]	Clustering Based Prioritization (CBP) technique.	Reduced	Improved	-
4.	Miao, Y. et al., 2013 [16]	A clustering based strategy	Reduced	Improved	-
5.	Kanimozhi, R., & Balakrishnan, J. R. 2014 [15]	Cosine Similarity based clustering	-	Improved	-
6.	Rani, N., & Chaudhary, J. 2015 [17]	A clustering improved cost effective approach	Reduced	-	Improved
7.	Dobuneh, M. R. N., et al., 2015[9]	A Self Organizing Maps (SOMs) based test case prioritization method	-	Improved	-
8.	Verma, A. et al., 2016 [19]	K- means algorithm	Reduced	-	Improved
9.	Haraty, R. A. et al., 2016[10]	Test case prioritization method	-	Improved	Improved
10.	Chang A S , et al., 2018 [5]	Adaptive Partition Testing	Reduced	Improved	-
11.	Chen, J. et al., 2018[4]	Adaptive Random Sequence (ARS) approach based on clustering methods	-	Improved	Improved
12.	Arnaldo M S, (2018), [3]	Applying Feedback Information for Random Partition Testing	Reduced	Not Improved	Not Improved

Previous studies have emphasized the various testing techniques with clustering algorithms that are effective with respect to the time required, the test Fault Detection and code coverage. Implementation of any type of ARPT family technique requires consideration of certain parameters mentioned above. On applying ARPT with clustering methods contributes towards the improvement of techniques.

3 IMPROVED ADAPTIVE RANDOM PARTITIONING

ARPT is the combination of AT and RPT strategies. The AT testing strategy utilizes the testing history which is collected online. [8] The RPT testing strategy is randomly partitioning the test cases in a test suite to test software. The proposed IART testing strategy uses the test cases which are clustered using different clustering methods instead of randomly partitioning test cases in RPT [8]. The following clustering methods are used to cluster the test cases:

- Expectation maximization (EM) algorithm
- K- means algorithm
- Non-negative Matrix Factorization (NMF)
- Self-organizing map (SOM) clustering.

3.1Expectation Maximization for ARPT

The EM algorithmic rule [1] is employed to find (locally) maximum likelihood parameters of a statistical model and it is often employed in cases wherever the matter can't be resolved directly by the equations. In addition, it includes unknown parameters and well-known knowledge comments with latent variables. Naturally discovering a maximum likelihood solution needs captivating the derivatives of the likelihood function with regard to all the unknown values, the parameters, and the latent variables and concurrently resolving the resulting equations. The EM clustering algorithm is used for partition the test cases. The key aim of this algorithm is to maximize the objective function $'L(T|\theta)'$ which measures the goodness of the partitioning. According to a probability distribution defined by the parameters for the mixture which is denoted by $'\theta'$ the partitioning is created. A mixture of multivariate Bernoulli distributions are used for the probability distribution representation and it is written as follows:

$$P(T|w_i;\theta) = \left(\prod_{term_m \in T} q_{mi}\right)\left(\prod_{term_m \notin T}(1-q_{mi})\right) \quad (1)$$

Equation (1), $\theta = \{\theta_1, \theta_2, \dots, \theta_I\}, \theta_i = (\alpha_K, q_{1K}, \dots q_{MI})$ and $q_{mi} = P(U_m = 1|w_i)$, U_m is the random variable for the Bernoulli Naïve Bayes model. $P(U_m = 1|w_i)$ is the probability which a test case from cluster $'w_i'$ contains term $term_m$. α_i denotes the prior of cluster w_i: the probability that a

test case 'T' is in $'w_i'$ on the off chance that it has no data about T. The mixture model is given as follows:

$$P(T|\theta) = \sum_{i=1}^{I} \alpha_i \left(\prod_{term_m \in T} q_{mi}\right)\left(\prod_{term_m \notin T}(1 - q_{mi})\right) \quad (2)$$

In the above described model, a test case is created by first picking a cluster 'i' with probability $'\alpha_i'$ and then create terms of the test cases according to the parameters q_{mi}. The maximization step of the EM algorithm recalculates the conditional parameter q_{mi} and the priors are described as follows:

$$q_{mi} = \frac{\sum_{j=1}^{J} r_{ji} I(term_m \in T_j)}{\sum_{j=1}^{J} r_{ji}} \quad \alpha_i = \frac{\sum_{j=1}^{J} r_{ji}}{J} \quad (3)$$

Equation (3), $I(term_m \in T_j) = 1$ if $term_m \in T_j$ else $I(term_m \in T_j) = 0$, r_{ji} denotes the soft assignment of test case T_j to cluster j as computed in the preceding iterations. These are the maximum likelihood estimates maximize the likelihood of the data given the model. The expectation step calculates the soft assignment of test cases to clusters given the current parameters q_{mi} and α_i which is given as follows:

$$r_{ji} = \frac{\alpha_i (\prod_{term_m \in T} q_{mi})(\prod_{term_m \notin T}(1-q_{mi}))}{\sum_{i=1}^{I} \alpha_i (\prod_{term_m \in T} q_{mi})(\prod_{term_m \notin T}(1-q_{mi}))} \quad (4)$$

The expectation step applies equation (1) and (2) to calculate the likelihood that w_i created test cases T_j.

EM algorithm for ARPT

Input: Collection of unlabelled test cases
Output: Clusters of test cases
Step 1: Calculate the probability distribution using multivariate Bernoulli distributions which is computed by using equation 1.
Step 2: Calculate the conditional parameter and the priors using equation 3 (E-Step).
Step 3: Calculate the soft assignment of test cases which cluster the test cases using equation 4 (M-Step).

3.2 K-means for ARPT

K-means is one among the smallest amount troublesome unsupervised learning algorithms that solve the well-known clustering drawback [2]. The system takes when a simple and easy approach to group a given informational collection through a particular range of bunches (expect 'k' groups) settled from the earlier. The first thought is to characterize 'k' centroids, one for every cluster. These centroids ought to be placed in a very cunning manner owing to various area causes distinctive outcome. So, the better alternative is to put them the maximum amount as doable secluded from one another. Succeeding step is to require every purpose of happiness to a piece of given information set and associate it to the closest centroid. Once no purpose is unfinished, the primary step is completed and an early cluster age is finished. At now knew centroids are had to be compelled to compute as barycenter's of the clusters ensuing from the previous step. At the moment k new centroids are obtained, a brand new binding has got to be done between constant information set points and also the nearest new center of mass. A loop has been generated. As a result of this loop, it should notice that the k centroids modification their location step by step till no additional changes are done. In alternative words, centroids don't move to any extent further. In this research work, the K-means clustering is used to partition the test cases by grouping similar type test cases based on the similarity measure. Before clustering the similar type of test cases in a group, the term count is calculated. Term count is the sum of occurrence of terms in the test cases. Then, a similarity measure needs to be calculated between each test case. This measure helps in identifying the degree of similarity or difference between the test cases. The similarity or dissimilarity between a pair of test cases can be defined explicitly or implicitly which plays an important role in clustering. The similarity between test cases is calculated by using cosine similarity. It can be calculated by using following equation

$$\cos(T_i, T_j) = \frac{T_i . T_j}{||T_i|| ||T_j||} \quad (5)$$

The cosine estimations of the test cases are in the ranges from 0 to 1. In view of the cosine similitude the If the cosine value returns 1, then the test cases $'T_i'$ and $'T_j'$ at that point the test cases are like each other else it is unique Based on the cosine similarity the test cases are partitioned.

K-means algorithm for ARPT

Input: Collection of unlabelled test cases
Output: Clusters of test cases
Step 1: Select k test cases as initial centers
Step 2: Calculate the term count
Step 3: Calculate the cosine similarity of test cases with the centers
Step 4: Assign each test case to the maximum similarity center
Step 5: Recalculate the centers of each clusters
Step 6: Repeat step 4 and 5 until distribution of test cases in clusters do not change

3.3 Non-Negative Matrix Factorization (NMF) for ARPT

Non-negative matrix factorization (NMF) [11], also Non-negative matrix approximation could be a cluster of algorithms in statistical procedure and algebra wherever a matrix 'V' is factorized into (usually) 2 matrices 'W' and 'H', with the property that every one 3 matrices don't have any negative parts. This non-negativity makes the ensuing matrices easier to examine. Also, in applications like process of audio spectrograms non-negativity is inherent to the info being thought-about. Since the matter isn't precisely soluble normally, it's normally approximat ed numerically. The approximate non-negative matrix factorization: the matrix 'V' is depicted by the 2 smaller matrices W and H, which, once increased, roughly reconstruct 'V'. NMF as a probabilistic graphical model: visible units (V) are connected to hidden units (H) through weights 'W', so 'V' is produced from a likelihood distribution with mean. For the test case partitioning, term frequency vector is used to represent the test cases. The term frequency vector F_i of test case T_i is represented as follows:

$$F_i = [f_{1i}, f_{2i}, \dots, f_{mi}]^T \quad (6)$$

Equation (6) $'F_i'$ is normalized to unit Euclidean length. Using F_i as i-th column the $m \times n$ matrix is constructed term-test case matrix is constructed. This matrix is utilized to conduct the non-negative factorization. Moreover the test case clustering will be directly obtained from the factorization result. Assume that the given test cases consists of 'k' test case clusters. The main intend is to factorize F into the non-negative $m \times k$ matrix 'U' and the non-negative k$\times n$ matrix $'V^T'$ that minimizes the following objective function

$$J = \frac{1}{2}||F - UV^T|| \quad (7)$$

NMF decomposition is given as follows:

$$F = UV^T \quad (8)$$

Let, $U = [u_{ij}], V = [v_{ij}], U = [U_1, U_2, \dots U_k]$, u_{ij} and v_{ij} is represented as follows:

$$u_{ij} \leftarrow u_{ij} \frac{(FV)_{ij}}{(UV^TV)_{ij}} \quad (9)$$

$$v_{ij} \leftarrow v_{ij} \frac{(F^TU)_{ij}}{(VU^TU)_{ij}} \quad (10)$$

Equation (8) and (9) u_{ij} denotes the i-th row 'j'-th column element of 'U', v_{ij} denotes the i-th row and j-th column of 'V' and $'F_{ij}'$ denotes the i-th row and 'j'-th column of 'F'.

The objective function 'J' is none increasing under the above updating formulas (9 and 10) and that the convergence of the iteration is guaranteed. It should be noted that the solution to minimizing the criterion function 'J' is not unique. To make solution unique, it is required that the Euclidean length of the column vector in matrix 'U' is one. The normalizing 'U' can be obtained by using following equation:

$$v_{ij \leftarrow} v_{ij} \sqrt{\sum_i u_{ij}^2} \quad (11)$$

$$u_{ij} \leftarrow \frac{u_{ij}}{\sqrt{\sum_i u_{ij}^2}} \quad (12)$$

Each element u_{ij} of matrix 'U' denotes the degree to which term belongs to cluster 'j' and each element v_{ij} of matrix 'V' denotes to which degree test case 'i' is associated with the cluster 'j'. If the test case solely belongs to cluster then v_{ix} will take on a large value while rest of the elements in i-th row vector of 'V' will take on a small value close to zero.

NMF Algorithm for ARPT

Input: Collection of unlabelled test cases

Output: Clusters of test cases

Step 1: Construct the term-test case matrix F in which column i denotes the weighted term frequency vector of test case T_i

Step 2: Perform NMF on F to obtain U and V non-negative matrices by using equation 9 and 10

Step 3: Normalize U and V using equation 11 and 12

Step 4: Utilize matrix V to determine the cluster label of each test cases. Examine each row of i of matrix V. Assign test case T_i to cluster x if $x = \arg \max_j v_{ij}$

3.4 Self-Organizing Map for ARPT

The self-organizing map (SOM) [7] is outlined as a neural technique for grouping. They are totally different from alternative artificial neural networks as they relate competitive learning as opposition error-correction learning, and within the sense that they use a section perform to preserve the properties of the input space. It's demonstrating the 2 abstraction areas of link among clusters. SOM has been ready to gift the data points that are in one or three-dimensional house, provided by SOM capabilities. Moreover, because of the simple of visualization and therefore the trade-off between info content two-dimensional areas are used more often. Initially, the term frequency within the test cases is calculated that is given as input vectors to SOM technique. The weights and learning rates are initialized. Once the term counts are given, supported the initial weights, the winner unit is computed

by mistreatment the euclidian distance methodology. Supporte d the winner unit choice, the weights are updated for that exact winner unit competitive learning rule. The subsequent algorithmic rule defines the action partitioning process:

SOM Algorithm for ARPT

Input: Collection of unlabelled test cases, $iteration_{max}$, $term_{freq}$

Output: Cluster of test cases

Step 1: Initialize topological neighborhood parameters and learning rate initial weights

Step 2: while (i<$iteration_{max}$) do

Step 3: for (j=1; j<n; j++) //n is the number of test cases

Step 4: Compute the squared Euclidean distance for each test cases by using following equation

$$D(j) = \sum (w_{mj} - x_j)^2; m = 1,2, \dots n, j = 1,2, \dots n$$

//x is the input test case

Step 5: Find test cases when D (j) is minimum

Step 6: For all test cases, with specified neighborhood of each test case and for all m update the weights by using following equation

$$w_{mj(new)} = w_{mj(old)} + \propto [x_j - w_{mj(old)}]$$

Step 7: Update the learning rate

Step 8: Reduce the radius of topological neighborhood at specified times n.

The test cases which are clustered by using EM, K-means, NMF and SOM algorithm are used in RPT which improves the defect detection efficiency of ARPT-1 and ARPT-2 testing strategies.

4 CASE STUDY

The case study of this proposed research work is carried out in three different software called univocityparser, marc4j and jsoup. The univocity-parser is a suite of extremely fast and reliable parsers for Java. It provides a consistent interface for handling different file formats, and a solid framework for the development of new parsers. The marc4j is software that provides an easy to use Application Programming Interface (API) for working with MARC and MARCXML in Java. jsoup is a Java library for working with real-world HTML. It provides a very convenient API for extracting and manipulating data, using the best of DOM, CSS, and jquery-like methods.

Each software is described in the following Table 1.

TABLE 2

SOFTWARE DESCRIPTION

Software	No. of test cases	No. of programs
univocityparser	1000	137
marc4j	1000	93
jsoup	1000	56

5 RESULT AND DISCUSSION

The experiments results are conducted in order to prove the effectiveness of the proposed Clustering algorithms in ARPT testing strategies. The comparison is based on the time consumption, defect detection efficiency, number of test cases and code coverage of both existing and proposed techniques.

5.1 Time Consumption

Time consumption is the amount of time taken to test software. The following Table 2 shows the comparison of time consumption between the existing ARPT-1 testing strategy and proposed ARPT-1 with clustering methods based testing

strategy.

TABLE 3
COMPARISON OF TIME CONSUMPTION FOR ARPT-1 AND ARPT-1 WITH CLUSTERING ALGORITHMS

Testing Strategies	*Software*		
	Univocityparser	*Marc4j*	*Jsoup*
ARPT-1	*2.5*	*2.49*	*2.48*
ARPT-1-EM	*2.45*	*2.43*	*2.42*
ARPT-1-NMF	*2.4*	*2.41*	*2.42*
ARPT-1-Kmeans	*2.38*	*2.33*	*2.32*
ARPT-1-SOM	*2.37*	*2.3*	*2.29*

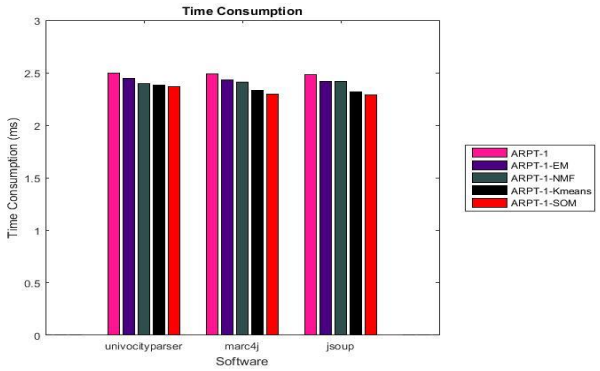


Fig.1*. Comparison of Time Consumption for ARPT-1 and ARPT-1 with Clustering algorithms*

Figure 1 shows the comparison of time consumption for existing ARPT-1 and proposed ARPT-1-EM, ARPT-1-NMF, ARPT-1-Kmeans and ARPT-1-SOM. X axis denotes the three different software are univocityparser, marc4j and jsoup and Y axis denotes the time consumption in milliseconds. From Figure 1 it is proved that the proposed ARPT-1-SOM has better time consumption than the other testing strategies for univocityparser, marc4j and jsoup software.
Table 4 shows the comparison of time consumption between existing ARPT-2 testing strategy and proposed ARPT-2 with clustering methods based testing strategy.

TABLE 4
COMPARISON OF TIME CONSUMPTION FOR ARPT-2 AND ARPT-2 WITH CLUSTERING ALGORITHMS

Testing strategies	Software		
	Univocityparser	marc4j	jsoup
ARPT-2	2.9 ms	2.88ms	2.89ms
ARPT-2-EM	2.85ms	2.83ms	2.82ms
ARPT-2-NMF	2.81ms	2.79ms	2.78ms
ARPT-2-Kmeans	2.73ms	2.72ms	2.73ms
ARPT-2-SOM	2.67ms	2.66ms	2.64ms

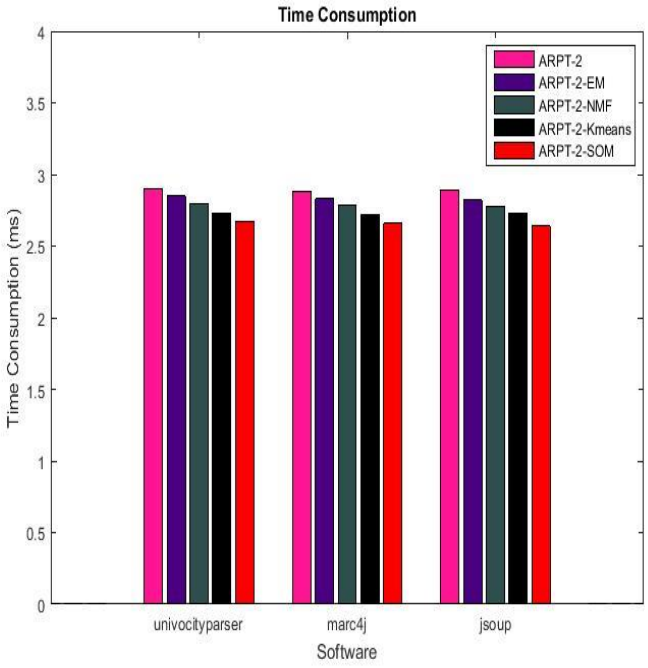


Fig. 2*. Comparison of Time Consumption for ARPT-2 and ARPT-2 with Clustering algorithms*

Figure 2 shows the comparison of time consumption for existing ARPT-2 and proposed ARPT-2-EM, ARPT-2-NMF, ARPT-2-Kmeans and ARPT-2-SOM. X-axis denotes the three different software are univocityparser, marc4j and jsoup and Y-axis denotes the time consumption in milliseconds. From Figure 2, it is proved that the proposed ARPT-2-SOM has better time consumption than the other testing strategies for univocityparser, marc4j and jsoup software.

5.2 Defect Detection Efficiency
Defect detection efficiency (DDE) is the number of defects detected during a phase/stage that are injected during that same phase divided by the total number of defects injected during that phase. It can be calculated by using following formula:[20]

$$DDE = \frac{No.of\ defects\ injected\ AND\ Detected\ in\ a\ phase}{Total\ No.of\ Defects\ injected\ in\ that\ phase} \times 100\%$$

The following Table 5 shows the comparison of time consumption between existing ARPT-1 testing strategy and proposed ARPT-1 with clustering methods based testing strategy.

TABLE 5
COMPARISON OF DEFECT DETECTION EFFICIENCY FOR ARPT-1 AND ARPT-1 WITH CLUSTERING ALGORITHMS

Testing strategies	Software		
	Univocityparser	marc4j	jsoup
ARPT-1	68.95 %	69.21%	71.15%
ARPT-1-EM	70.38%	71.57%	72.90%
ARPT-1-NMF	73.36%	72.52%	73.82%
ARPT-1-Kmeans	73.46%	72.82%	73.84%
ARPT-1-SOM	73.97%	73.85%	74.20%

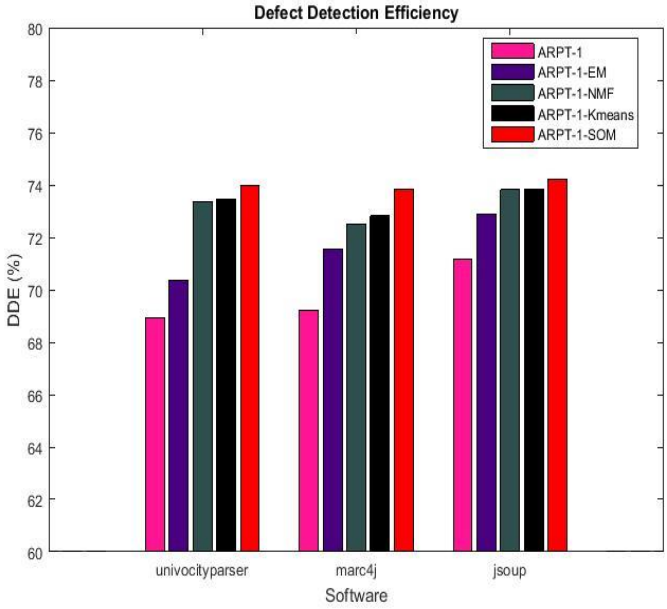


Fig. 3*.Comparison of Defect Detection Efficiency for ARPT-1 and ARPT-1 with Clustering algorithms*

Figure 3 shows the comparison of defect detection efficiency for existing ARPT-1 and proposed ARPT-1-EM, ARPT-1-NMF, ARPT-1-Kmeans and ARPT-1-SOM. X-axis denotes the three different software are univocityparser, marc4j and jsoup and Y-axis denotes the defect detection efficiency in (%). From Figure 3, it is proved that the proposed ARPT-1-SOM has better defect detection efficiency than the other testing strategies for univocityparser, marc4j and jsoup software. The following Table 6 shows the comparison of time consumption between existing ARPT-2 testing strategy and proposed ARPT-2 with clustering methods based testing strategy.

TABLE 6
COMPARISON OF DEFECT DETECTION EFFICIENCY FOR ARPT-2 AND ARPT-2 WITH CLUSTERING ALGORITHMS

Testing strategies	*Software*		
	Univocityparser	*marc4j*	*jsoup*
ARPT-2	*60.43 %*	*61.54%*	*61.37%*
ARPT-2-EM	*60.87%*	*61.98%*	*61.79%*
ARPT-2-NMF	*64.97%*	*65.13%*	*65.23%*
ARPT-2-Kmeans	*65.08%*	*65.15%*	*65.28%*
ARPT-2-SOM	*66.15%*	*66.19%*	*66.25%*

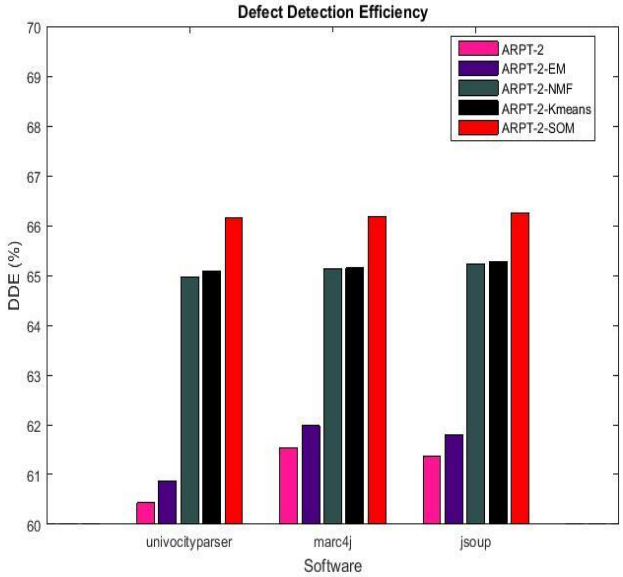


Fig. 4*. Comparison of Defect Detection Efficiency for ARPT-2 and ARPT-2 with Clustering algorithms*

Figure 4 shows the comparison of defect detection efficiency for existing ARPT-2 and proposed ARPT-2-EM, ARPT-2-NMF, ARPT-2-Kmeans and ARPT-2-SOM. X-axis denotes the three different software are univocityparser, marc4j and jsoup and Y-axis denotes the defect detection efficiency in (%). From Figure 4, it is proved that the

proposed ARPT-2-SOM has better defect detection efficiency than the other testing strategies for univocityparser, marc4j and jsoup software.

5.3 Code Coverage

Code coverage [6] determines how much code is being tested. It ensures to maintain the test quality over the life cycle of a project and to know how well testing strategy actually tests the code. It can be calculated based on a coverage matrix.

$$cov(T, C) = \frac{|c \in C | c \ covered \ by \ T|}{|C|}$$

where, T be the set of test cases selected and C be set of code elements.

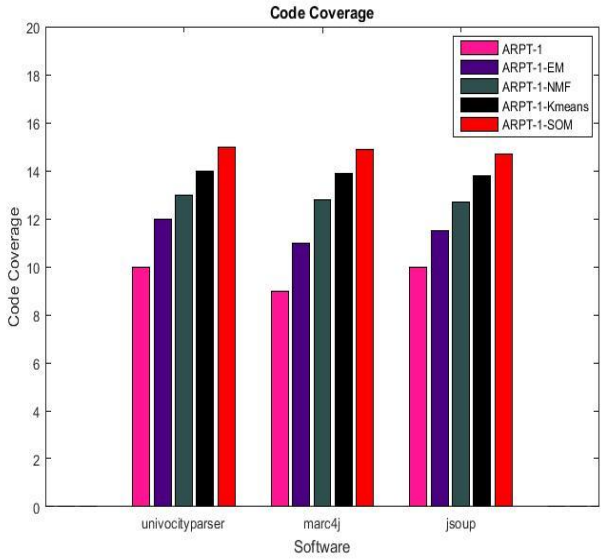


***Fig. 5**. Comparison of Code Coverage for ARPT-1 and ARPT-1 with Clustering algorithms*

Figure 5 shows the comparison of code coverage for existing ARPT-1 and proposed ARPT-1-EM, ARPT-1-NMF, ARPT-1-Kmeans and ARPT-1-SOM. The X-axis denotes the three different software are univocityparser, marc4j and jsoup and Y-axis denotes the code coverage. From Figure 5, it is proved that the proposed ARPT-1-SOM has better code coverage than the other testing strategies for univocityparser, marc4j and jsoup software. Figure 6 shows the comparison of code coverage for existing ARPT-2 and proposed ARPT-2-EM, ARPT-2-NMF, ARPT-2-Kmeans and ARPT-2-SOM. X-axis denotes the three different software are univocityparser, marc4j and jsoup and Y-axis denotes the code coverage. From Figure 6, it is proved that the proposed ARPT-2-SOM has better code coverage than the other testing strategies for univocityparser, marc4j and jsoup software.

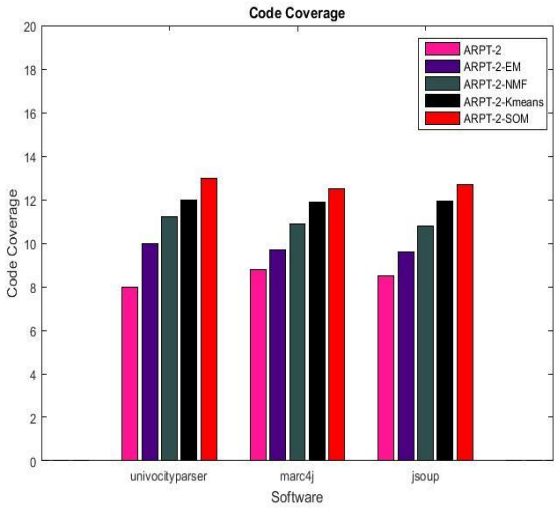


***Fig.6.** Comparison of Code Coverage for ARPT-2 and ARPT-2 with clustering algorithms*

*5.*4 Selected Test Cases

An efficient testing strategy detects the defects in software with minimum number of test cases and high coverage.

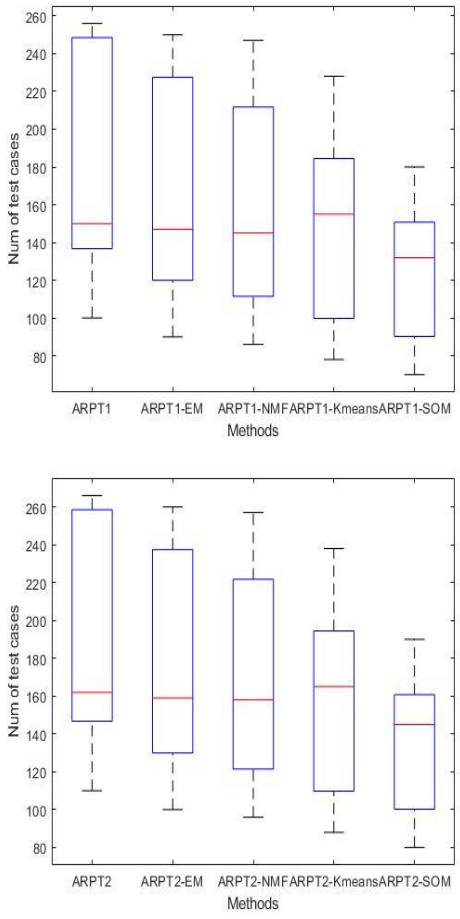
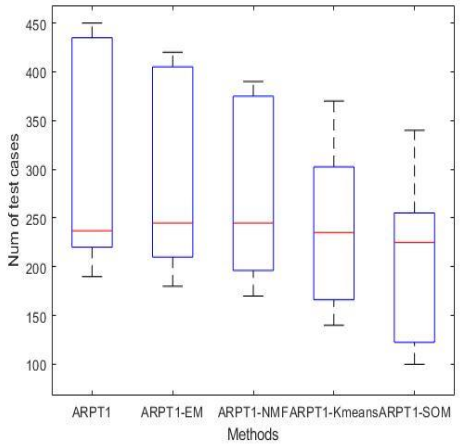
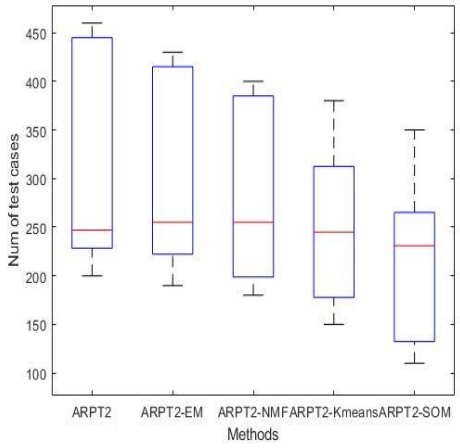


***Fig. 7.** Comparison of Selected test cases for univocityparser software, (a) ARPT-1 and ARPT-1 with clustering methods (b) ARPT-2 and ARPT-2 with Clustering algorithms*

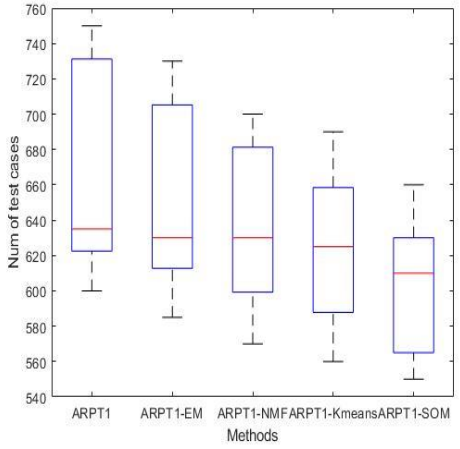


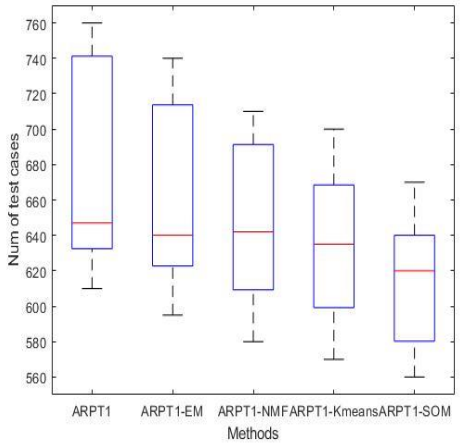


(a)

(b)

Fig.8*. Comparison of Selected test cases for marc4j software, (a) ARPT-1 and ARPT-1 with clustering methods (b) ARPT-2 and ARPT-2 with Clustering algorithms*





(a) (b)

Fig.9*. Comparison of Selected test cases for jsoup software, (a) ARPT-1 and ARPT-1 with clustering methods (b) ARPT-2 and ARPT-2 with Clustering algorithms*

Figure 7, 8 and 9 shows the number of test cases selected by ARPT-1, ARPT-1-EM, ARPT-1-NMF, ARPT-1-Kmeans, ARPT-1-SOM, ARPT-2, ARPT-2-EM, ARPT-2-NMF, ARPT-2-Kmeans and ARPT-2-SOM, testing strategies for univocityparser, marc4j and jsoup software. The red line indicates at that number of test cases the defects in the software is detected. From the results of selected test cases, it is proved that the ARPT-1-SOM and ARPT-2-SOM detects the defects with less number of test cases than the other testing strategies. From the above result and discussion, the performance of ARPT 1 and ARPT 2 is compared with ARPT with clustering methods. Finally the result driven is that the performance of ARPT 1 with SOM and ARPT 2 with SOM yields better results than the others.

6 CONCLUSION

Henceforth, rather than testing with the entire arrangement of test cases, a cluster is shaped so that a single representative from each profile will be an effective and efficient alternative. In this research work, a mining approach is used to have better test cases. The best test cases can be generated, selected and are used for testing. Mining the test suite will provide a better set of test cases. ARPT with the clustering algorithm automatically derives good parameter values for test application, and it is capable of achieving less time for executing a test of an application. Thus the proposed Improved ARPT (IARPT) solves the parameter space of the problem between the target method and objective function of the test data. It achieves high test case coverage within less time and produces better accuracy. Finally, the performance of the ARPT-1-SOM andARPT-2-SOM is better and defect detection is high and less time to coverage of all test cases rather than the other methods. In future ARPT 1 and ARPT2 can combinefor getting better results.

REFERENCES

[1] Avinash Kak. “Expectation-maximization algorithm for clustering multidimensional numerical data”, An RVL Tutorial Presentation at Purdue University. 2014.

[2] Aditi Anand Shetkar and S. Fernandes. “Text categorization of documents using K-means and K-means++ clustering algorithm”, International Journal on Recent and Innovation Trends in Computing Communication, vol. 4, no. 6, pp. 485-489, 2016.

[3] Arnaldo M S, “Applying Feedback Information for Random Partition Testing”. IEEE ICIS, 224-228. 2018

[4] Chen, J., Zhu, L., Chen, T. Y., Towey, D., Kuo, F. C., Huang, R., & Guo, Y.” Test case prioritization for object-oriented software: An adaptive random sequence approach based on clustering”. Journal of Systems and Software, 135, 107-125, 2018

[5] Chang A S ., Hepeng D ; Huai L., Tsong Y C., Kai-Yuan C. Adaptive Partition Testing. IEEE Transactions on Computers, 68(2), Page(s): 157 – 169, 2018.

[6] Dávid Tengeri, Árpád Beszédes, Dávid Havas and Tibor Gyimóthy. “Toolset and program repository for code coverage-based test suite analysis and manipulation”, IEEE 14th International Working Conference on Source Code Analysis and Manipulation, pp. 47-52, 2014.

[7] Dino Isa, V. P. Kallimani and Lam Hong Lee. “Using the self-organizing map for clustering of text documents”, Expert Systems with Applications, vol. 36, no. 5, pp. 9584-9591, 2009.

[8] K. Devika Rani Dhivya, V. S . Meenakshi. “Improved Time Performance of Adaptive Random Partition Software Testing by Applying Clustering Algorithms”, International Journal of Advanced Research in Computer Science, .8(5). 0976-5697, 2017.

[9] Dobuneh, M. R. N., Jawawi, D. N., & Vahidi, M.. “Clustering Test Cases in Web Application Regression Testing using Self-Organizing Maps”. Int. J. Advance Soft Compu. Appl, 7(2). 2015

[10] Haraty, R. A., Mansour, N., Moukahal, L., & Khalil, I.. “Regression Test Cases Prioritization Using Clustering and Code Change Relevance”. International Journal, 2016

[11] Hiroyuki Shinnou and Minoru Sasaki. “Refinement of document clustering by using NMF”, In Proceedings of the 21st Pacific Asia Conference on Language, Information and Computation, pp. 430-439, 2007.

[12] Upadhyay, A. K., & Misra, A. K. . “Prioritizing test suites using clustering approach in software testing”. International Journal of Soft Computing and Engineering (IJSCE), 2(4) , 2012.

[13] Upadhyay, A. K. & Misra, A. K..” Clustering based Prioritization of Test Suites in Software Testing”. International Journal of Scientific & Engineering Research, 3(8), 1-5, 2012.

[14] Junpeng Lv, Bei-Bei Yin and Kai-Yuan Cai. “On the asymptotic behavior of adaptive testing strategy for software reliability assessment”, IEEE transactions on Software Engineering, vol. 40, no. 4, pp. 396-412, 2014.

[15] Kanimozhi, R., & Balakrishnan, J. R., “Cosine Similarity based Clustering for Software Testing using Prioritization”. IOSR Journal of Computer Engineering (IOSR-JCE), 16(1), 75-80. 2014.

[16] Miao, Y., Chen, Z., Li, S., Zhao, Z., & Zhou, Y. A” clustering-based strategy to identify coincidental correctness in fault localization”. International Journal of Software Engineering and Knowledge Engineering, 23(05), 721-741, 2013.

[17] Rani, N., & Chaudhary, J. “A Clustering Improved Cost Effective Approach for Mutation Testing”. Procedia Computer Science, 58, 593-602., 2015

[18] Sapna, P. G., & Mohanty, H.” Clustering test cases to achieve effective test selection”. In Proceedings of the 1st Amrita ACM-W Celebration on Women in Computing in India (p. 15). ACM. 2010

[19] Verma, A., Bajaj, R., & Luthra, I. K..” A Density based K-means Clustering for Test Case Prioritization in Regression Testing Result-II”. International Journal of Computer Science and Technology (IJCST), 7(1), 114-116. 2016

[20] Zhang, X., Teng, X., & Hoang Pham. “Considering fault removal efficiency in software reliability assessment”. IEEE Transactions on Systems, Man, and Cybernetics , 33(1), 114-120, 2003.