Generating Optimized Pairwise Test Cases by using K-Means Algorithm

M. Lakshmi Prasad¹, M. Keerthi², K. Sai Srikar³, V. Divya⁴

¹Assoc Prof, Dept of CSE, NBKR Institute of Science & Technology, Nellore, AP, India, E-mail: prasad.hinduniv@gmail.com.
²Dept of CSE, NBKR Institute of Science & Technology, Nellore, AP, India, E-mail: keerthimannem550@gmail.com.
³Dept of CSE, NBKR Institute of Science & Technology, Nellore, AP, India, E-mail: srikar326@gmail.com.
⁴Dept of CSE, NBKR Institute of Science & Technology, Nellore, AP, India, E-mail: divyavelimreddy@gmail.com.

Abstract: Pair wise testing is an effective combinatorial test generation technique that can generate relative small test suite to cover all pairs of parameter values at least once. The varieties of techniques have been used for pairwise test suite generation by some researchers. In order to get optimized result, this paper proposes a hybrid optimization algorithm by using data mining techniques. Data mining finds similar patterns in test cases which helped us in finding out redundancy incorporated by automatic generated test cases. A series of experiments are conducted to evaluate the proposed algorithm. Experiment results show that the proposed algorithm is very competitive with respect to other approaches reported in the literature. In this paper, we habit the data mining approach of clustering technique i.e k-means and k-medoids are used in software testing to reduce the test suite to obtain the optimality and improve the efficiency of software testing.

Keywords: Software Testing, Test Cases, Test Data Generation, Data Mining; Test Case, Test Suite, Cluster.

I. INTRODUCTION

Testing software is a very important and challenging activity. The main objective of software testing is to eliminate as many errors as possible to ensure that the tested software meets an acceptable level of quality. The tests have to be performed within budgetary and scheduled limitations. An important activity in testing is test case design. Many programming groups are relying more and more on automated testing, it require a well-developed test suite of testing scripts in order to be truly useful. If the Test suites tend to grow in size as software evolves, then testing becomes too cost to execute entire test suites. The test suite reduction techniques significantly reduce the size of the test suites. A test suite is a collection of test cases for particular software. Redundancy of test cases will be possible in the software. Redundancy is the repetition of data, between one test case and the other. So it is obvious that the reasonable structure of test suite is one of the key points in software testing achieve by which lot of time can be saved from executing redundant or unnecessary test cases. This replicated data isn’t visible enough to capture unless and until the sophisticated techniques like data mining is used. In this paper, the proposed methodology of data mining technique is used with software testing to remove the redundant test case so that the test suites are reduced or minimized.

A. Test Suite

The main objective of software testing is how to select test cases with the aim of uncovering as many defects as possible. A test suite is run on the software under test and the output is examined by the tester, by comparing actual output with the expected output. If the output is incorrect then error has been discover. So, the program must be changed and testing must start again. A test suite is a collection of test cases that are intended to be used to test a software program to show that it has some specified set of behaviours [5]. Writing effective test cases for a application is a skill that can be achieved by some experience and in-depth study of the application on which test cases are being written [7].

B. Test Suite Reduction

Test suite reduction techniques try to remove redundant test cases of a test suite. The test suite minimization problem can be formally stated as follows.

Given:
- A test suite $T$ of test cases $\{t_1, t_2, t_3, \ldots, t_k\}$.
- A set of testing requirements $\{r_1, r_2, r_3, \ldots, r_n\}$ that must be satisfied to provide the desired testing coverage of the program.
- Subsets $\{T_1, T_2, T_3, \ldots, T_n\}$ of $T$, one associated with each of the $r_i$’s, such that any one of the test cases $t_j$ belonging to $T_i$ satisfies $r_i$.

Combinatorial Interaction Testing (CIT) systematically explores pairwise feature interactions inside a given system, by effectively combining all pair-tuples of parameter assignments in the smallest possible number of test cases. In particular, pairwise testing ($t = 2$), consists in testing all pairs of input values at least once. It has been empirically confirmed that pairwise testing can detect a significantly large part (typically 50% to 75%) of faults in a system, thus many CIT approaches have been developed and are currently applied in practice. CIT can be applied to a wide variety of

Copyright @ 2017 IJATIR. All rights reserved.
problems: highly-configurable software systems, software product lines, hardware systems, etc. In this paper, the application of data mining techniques with software testing is used for reducing the size of the test suite. The less, the number of test cases, the time taken for executing the program should also be less. This consequently improves the effectiveness of the test process.

II. LITERATURE SURVEY

K. Tsumura, et.al [1] had proposed a pairwise coverage-based testing with selected elements in a query for database applications. They present two new database applications using Plain Pairwise Coverage (PPC) and Selected Pairwise Coverage (SPC) for SQL queries called Plain Pairwise Coverage Testing (PPCT) and Selected Pairwise Coverage Testing (SPCT), respectively. These coverages are based on pairwise testing coverage, which employs selected elements in the SQL SELECT query as parameters. They also implement a coverage calculation tool and conduct case studies on two open source software systems. PPCT and SPCT can detect many faults, which are not detected by existing test methods based on predicates in the query. Furthermore, the case study suggests that SPCT can detect faults more efficiently than PPCT and the costs of SPCT can be further reduced by ignoring records filtered out by the conditions of the query. S. Nakornburi and T. Suwannasart, [2] had wished-for a tool for constrained pairwise test case generation using statistical user profile based prioritization," In this paper, They present a pairwise test generation tool called CPTG, a tool to generate test cases for pairwise testing by applying user profile for guiding and prioritizing in order to select optimal input parameters and values which do not depend on individual tester skills and also providing constraint handling solution between input parameters and values. They performed experiments and comparison with other tools. The experimental results of our tool demonstrated that our tool becomes particularly valuable in guiding testing with a maximized reliability by testing the most frequently used of the system and can generate comparable results of the size of the test case set.

A. B. Nasser, A. A. Alsewari and K. Z. Zamli, [3] had projected a Adopting search-based algorithms for pairwise testing. This paper presents a critical comparison of Search-Based algorithm for generating the pairwise test suite. An analysis of existing SB pairwise strategies shows the positive and negative points for each strategy thereby highlighting promising future directions in this area. H. L. Zakaria and K. Z. Zamli, [4] had defined Migrating Birds Optimization based strategies for Pairwise-testing. In this paper, They investigated the adoption of Migrating Birds Optimization (MBO) algorithm as a strategy to find an optimal solution for pairwise test data reduction. Two strategies have been proposed; the first strategy implements the basic MBO algorithm, called Pairwise MBO Strategy (PMBOS) and the second strategy implements an improved Pairwise MBO strategy, called iPMBOS. The iPMBOS enhances the PMBOS with multiple neighborhood structures and elitism. Based on the published benchmarking results, these two strategies offers competitive results with most existing strategies in terms of the generated test size. They also noted that iPMBOS outperforms PMBOS in several parameter configurations, especially when the test size generated is relatively, small.

R. Qi, Z. Wang, P. Ping and S. Li, [5] had projected a hybrid optimization algorithm for pairwise test suite-generation. In order to improve the performance of genetic algorithm, this paper proposes a hybrid optimization algorithm by augmenting genetic algorithm with two-stage hill climbing. The first stage is to improve all the individuals after genetic operations. The second stage is to improve the best solution of the current generation at the end of each generation. A series of experiments are conducted to evaluate the proposed algorithm. Experiment results show that the proposed algorithm is very competitive with respect to other approaches reported in the literature. R. E. Lopez-Herrejon, J. Ferrer, F. Chicanco, A. Egyed and E. Alba, [6] had developed a Comparative analysis of classical multi-objective evolutionary algorithms and seeding strategies for pairwise testing of Software Product Lines. In this paper, They study the application to SPL pairwise testing of four classical multi-objective evolutionary algorithms. They developed three seeding strategies - techniques that leverage problem domain knowledge - and measured their performance impact on a large and diverse corpus of case studies using two well-known multi-objective quality measures. Our study identifies the performance differences among the algorithms and corroborates that the more domain knowledge leveraged the better the search results. Our findings enable software engineers to select not just one solution (as in the case of single-objective techniques) but instead to select from an array of test suite possibilities the one that best matches the economical and technological constraints of their testing context.

C.B.A. L. Montero, L. A. V. Dias and A. M. d. Cunha, [7] had implemented a Case Study on Pairwise Testing-Application. In order to better understand it, this case study presents a comparison of an ad-hoc testing against pair wise testing applied under the same conditions. Ad-hoc and pair wise test cases were generated and performed on an industry application, and the pair wise tests were able to expose exactly the same defects as the ad-hoc tests, with roughly five times less effort. With these results, it is suggested that pair wise testing, combined with other techniques, skills, and knowledge on the business might be useful to reduce testing cost while increasing test-quality. S. Gao, B. Du, Y. Jiang, J. Lv and S. Ma, [8] had presented an efficient algorithm for pairwise test case generation in presence of constraints. In this paper, they present an effective algorithm, called IPO SAT (In-Parameter-Order-Satisfiability), for pairwise test case generation in presence of constraints. In their strategy, constraints are denoted as forbidden tuples, which are converted to conjunctive normal form. Then, the combination test cases which meet the constraints are found out by calling Boolean satisfiability (SAT) solvers. Besides, an optimization upon the process is given, in order to
Generating Optimized Pairwise Test Cases by using K-Means Algorithm

R. K. Sungkur and H. Muhamodsaroar,[9] had implemented a PWTool, A novel automated tool for pairwise testing. This paper investigates the suitability of pair-wise testing in automated testing and demonstrates how the In-Parameter Order (IPO) algorithm used in the generation of pair-wise combinations can be optimised. PWTool, a novel web-based application for generating pairwise test cases mainly for online systems is introduced. T. Sekizawa and T. Kotorii,[10] had projected a Case Study: Verification of Specifications of an Embedded System and Generation of Verification Items Using Pairwise Testing. In our study, they set conversion rules from specifications to formal models. Part of the conversion is done by hand in this study. Manual generation limits the preparation of individual evaluation items. To overcome this limitation they present an approach for automatically generating combinations of parameters for verification by applying the pair-wise testing method. Finally, they present experimental results. Note that the application of formal techniques, in this setting, is still in its preliminary stages. It is intended to develop formal techniques to the point where products may be automatically verified.

Kartheek Muthyala and Rajshekar naidu P.[11] had proposed “A novel approach to test suite reduction using data mining”. Their goal is to reduce the time spent in testing by reducing the number of test cases. For this they have incorporated data mining techniques to reduce the number of test cases. Data mining finds similar patterns in test cases which helped us in finding out redundancy incorporated by automatic generated test cases. They proposed a methodology based on clustering by which we can significantly reduce the test suite. The final test suite is tested for coverage which yielded good results. Mrs. B. Subashini and Dr. D. JeyaMala.[12] had proposed “Reduction of Test Cases Using Clustering Technique”. In this paper, the goal is to reduce the time spent in testing by reducing the number of test cases. For this the data mining approach of clustering technique is used in software testing to reduce the test suite. Mining of test case will improve the efficiency of software testing.

III. EXISTING SYSTEM

This section first explains the different coverage criteria, normally associated with combination strategies and then briefly describes the combination strategies that were identified in the literature. The combination strategies have been organized into different classes based on the amount of randomness of the algorithm and according to how the test suites are created. Fig.1 shows an overview of the classification scheme. The combination strategies labeled non-deterministic all depend to some degree on randomness.

A property of these combination strategies is that the same input parameter model may lead to different test suites. The simplest non-deterministic combination strategy is pure random selection of test cases. The group of non-deterministic combination strategies also includes two heuristic methods, CATS and AETG.

![Fig.1. Classification Scheme for Combination Strategies.](image)

IV. PROPOSED SYSTEM

Data Mining, also popularly known as Knowledge Discovery in Databases (KDD). Data mining an interdisciplinary subfield of computer science, is the computational process of discovering patterns in large data sets involving methods at the intersection of artificial intelligence, machine learning, statistics, and database systems. The overall goal of the data mining process is to extract information from a data set and transform it into an understandable structure for further use. Cluster analysis or clustering is the task of grouping a set of objects in such a way that objects in the same group are more similar to each other than to those in other groups. It can be achieved by various algorithms that differ significantly in their notion of what constitutes a cluster and how to efficiently find them. Popular notions of clusters include groups with small distances among the cluster members, dense areas of the data space, intervals or particular statistical distributions. Here, Clustering is one of the techniques which we are going to use for mining the test cases. The Algorithm K-means is one of the simplest unsupervised learning algorithms that solve the well-known clustering problem. The procedure follows a simple and easy way to classify a given data set through a certain number of clusters (assume k clusters) fixed a priori.

The main idea is to define k centroids, one for each cluster. These centroids should be placed in a cunning way because of different location causes different result. So, the better choice is to place them as much as possible far away from each other. The next step is to take each point belonging to a given data set and associate it to the nearest centroid. When no point is pending, the first step is...
completed and an early group age is done. At this point we need to re-calculate k new centroids as barycenter’s of the clusters resulting from the previous step. After we have these k new centroids, a new binding has to be done between the same data set points and the nearest new centroid. A loop has been generated. As a result of this loop we may notice that the k centroids change their location step by step until no more changes are done. In other words centroids do not move any more. Finally, this algorithm aims at minimizing an objective function, in this case a squared error function. The objective function

\[ J = \sum_{j=1}^{k} \sum_{i=1}^{n} \| x_i - c_j \|^2 \]  

(1)

where \( \| x_i - c_j \|^2 \) is a chosen distance measure between a data point and the cluster center, is an indicator of the distance of the n data points from their respective cluster centres.

A. Algorithmic Steps For K-Means Clustering

Let \( X = \{x_1, x_2, x_3, \ldots, x_n\} \) be the set of data points and \( V = \{v_1, v_2, \ldots, v_c\} \) be the set of centers.

- Randomly select ‘c’ cluster centers.
- Calculate the distance between each data point and cluster centers.
- Assign the data point to the cluster center whose distance from the cluster center is minimum of all the cluster centers.
- Assign the data point to the cluster center whose distance from the cluster center is minimum of all the cluster centers.
- Recalculate the new cluster center using:

\[ v_i = \left( \frac{1}{c_i} \right) \sum_{j=1}^{c_i} x_j \]  

(2)

where, ‘ci’ represents the number of data points in ith cluster.
- Recalculate the distance between each data point and new obtained cluster centers.
- If no data point was reassigned then stop, otherwise repeat from step 3.

One of the most widely used clustering algorithm is K-Means clustering. This minimizes the mean squared Euclidean distance from each data point to its nearest centre. Here we have a good control upon the number of clusters produced. So while reducing the test suit depending upon the number of test cases we wanted we can fix the value of k. Given a database \( D = \{t_1, t_2, \ldots, t_n\} \) and an integer value k, the k-Means algorithm defines a mapping \( F: D \to \{1, 2, \ldots, k\} \) (2). The design of the proposed system is as shown below. The system after detailed analysis has been identified to be presented with the following modules: Input module it takes the inputs as number of parameters and the number of values for those parameters for generating the test cases. Internal Module presents a data mining technique that is being implemented for generating the optimized pairwise test cases as shown in Fig.2. Initially this module, the hybrid technique takes the number of parameters and their values as input and sends the optimized result to the output module and finally the output module displays the optimal pairwise test suite as output.

Fig.2. Design of the Implementation System

A. Algorithm

Step 0: Begin

Step 1: Read the number of parameters

Step 2: Read the number of values for each parameter

Step 3: Read the values for those parameters

Step 4: See the initial parameter and domain values matrix as shown in table I

Step 5: Apply k-means Algorithm

- Consider the initial combination of parameter input pairs i.e. AB, AC
- Calculate the Euclidian distance values for all combination of parameters
- Form the clusters based on Euclidian distance value.
- Consider the smaller Euclidian distance value of combined parameter and insert the corresponding values of those combined parameter in to the test suite matrix.

Step 6: Similarly repeat the procedure of step 5.1 for the remaining combination of parameters until the all the pair is to be filled.

Step 7: Print the Optimized test suite Matrix

Step 8: Stop

The optimized pairwise test cases generated by using our strategy are as shown in the table I.

**TABLE I:**

<table>
<thead>
<tr>
<th>Test Vector Number</th>
<th>Element-A</th>
<th>Element-B</th>
<th>Element-C</th>
<th>Element-D</th>
</tr>
</thead>
<tbody>
<tr>
<td>t1</td>
<td>a0</td>
<td>b0</td>
<td>c0</td>
<td>d0</td>
</tr>
<tr>
<td>t2</td>
<td>a0</td>
<td>b1</td>
<td>c0</td>
<td>d1</td>
</tr>
<tr>
<td>t3</td>
<td>a1</td>
<td>b0</td>
<td>c1</td>
<td>d0</td>
</tr>
<tr>
<td>t4</td>
<td>a1</td>
<td>b1</td>
<td>c1</td>
<td>d1</td>
</tr>
<tr>
<td>t5</td>
<td>a2</td>
<td>b0</td>
<td>c2</td>
<td>d0</td>
</tr>
<tr>
<td>t6</td>
<td>a2</td>
<td>b1</td>
<td>c2</td>
<td>d1</td>
</tr>
<tr>
<td>t7</td>
<td>a0</td>
<td>b1</td>
<td>c3</td>
<td>d0</td>
</tr>
<tr>
<td>t8</td>
<td>a1</td>
<td>b0</td>
<td>c3</td>
<td>d1</td>
</tr>
</tbody>
</table>

V. RESULTS

The numbers of test cases generated for different systems which can be defined using various parametric combinations are shown in the table II.
Generating Optimized Pairwise Test Cases by using K-Means Algorithm

<table>
<thead>
<tr>
<th>TABLE II: Parametric Configurations Of Systems Considered For Testing</th>
</tr>
</thead>
<tbody>
<tr>
<td>Serial Number</td>
</tr>
<tr>
<td>---------------</td>
</tr>
<tr>
<td>1</td>
</tr>
<tr>
<td>2</td>
</tr>
<tr>
<td>3</td>
</tr>
<tr>
<td>4</td>
</tr>
<tr>
<td>5</td>
</tr>
<tr>
<td>6</td>
</tr>
<tr>
<td>7</td>
</tr>
</tbody>
</table>

From the above table, it is seen that the number of test cases generated by our technique is quiet minimal for considering any of the system configurations.

<table>
<thead>
<tr>
<th>TABLE III: Pair Wise Test Set Sizes For New Strategy And Other Algorithms</th>
</tr>
</thead>
<tbody>
<tr>
<td>System</td>
</tr>
<tr>
<td>----------------</td>
</tr>
<tr>
<td>All Pairs</td>
</tr>
<tr>
<td>AETF</td>
</tr>
<tr>
<td>PICT</td>
</tr>
<tr>
<td>SBC</td>
</tr>
<tr>
<td>PTSG_k</td>
</tr>
</tbody>
</table>

VI. CONCLUSION
Hence we concluded that an optimization technique was implemented to generate the optimized pairwise test cases for testing which helps in many applications. We took help of data mining and proposed a new technique in software testing which reduced the size of the test suite significantly. We have implemented test generation algorithm and have shown some experiential results. The experiment results illustrated that our proposed heuristic technique will yields better results when comparing to existing techniques like All pairs, IPO etc.. Our proposed strategy presented in this paper can be easily extended for n-way testing. We are investigating possible improvements of our algorithm without increasing time complexity.

VII. REFERENCES