PERFORMANCE STUDY OF COMBINATORIAL TESTING TO IMPROVE SOFTWARE SYSTEM QUALITY

M.BHARATHI, DR. V. SANGEETHA

Abstract: Combinatorial testing is mainly focus on the fault detection due to the small amount of input parameters. Pair-wise testing is the combinatorial testing approach of software system that tests every feasible combination of all pair of input parameter values. Software development is the process of intercommunicating components involved in generating and preserving the applications and frameworks resulting in a software product. An optimal selection is very useful to achieve the significant decrease in the count of interactions without any loss of fault detection ability. However, the time complexity is the major issue occurred during the interaction count. The study helps to decrease the time complexity involved during the count of interactions. In addition, it also detects the faults in software-under-test by reducing the number of test cases and hence improves the software system quality.

Keywords: Combinatorial Testing, Optimal Selection, Pair-wise Testing, Software Development.

1. Introduction: Combinatorial testing is not only minimizes the test cost but also improves the value of software application. It has the ability to verify that the significant in software system where the faults arise from the interactions of parameter. Combinatorial testing is also identifying the interactions between parameters that preserve the trigger faults rather than localize it. If the parameter localization is done in an effective manner, the software diagnosing and testing process are improved consecutively.

The main objective of combinatorial testing remains in the fault recognition due to the interaction of a lesser amount of input parameters. With the large number of probable combinations to be tested, testing systems through many variables is more expensive and involves time consuming process. In a system with enormous input parameters, it has the ability to check errors during the interactions between various input parameters. Several criteria's are available to integrate the test data and one of them is pair wise testing. In pair-wise testing, a combinatorial testing technique is essential to tests all the possible mixture of each pair of input parameter values.

Software development is subjected to cost and also the software testing is presumed as a time consuming process, where the cost and time spent on software development is increasing frequently. Most of the time, software are very necessary to deliver the lack of tests due to marketing pressure and also reduce the time and expense involved during the testing process. Software development is a framework in which they are employed to design and manage the method of increasing the information systems. Therefore, the test case minimization plays a significant role in decreasing the test cases without compromising their quality.

This paper is organized as follows: Section II discusses combinatorial testing in software system. Section III describes the existing combinatorial testing in software system technique, Section IV identifies the possible comparison between them, Section V explains the limitations as well as the related work and Section VI concludes the paper, research work is given as to optimally reduce the time consumption by minimizing the test size and also decrease the cost in an efficient manner.

2. Literature Review: Pairwise Test Set Generator using Genetic Algorithm (PTSG-GA) in [6] employed to offers the test set for pair-wise testing. But, it does not design to incorporate the constraint handling feature. Simplified Swarm Optimization (SSO) in [16] provides the optimized test suite by using Event-Interaction Graph (EIG). The test suites are very useful to remove the unnecessary event sequences. However, the issues of conventional PSO are that the convergence speed decreases as the number of iteration is increased in which they reduced to attain the better results. Combinatorial test generation technique in [18] based on one-test-at-a-time approach that converts the generation of all new test cases into linear pseudo-Boolean optimization issues. However, the approach is very complex to detect better parameter. Combinatorial Benders' Cuts (CBC) algorithm in [2] analyzed to develop the efficiency and lesser operation cost. Though, even it cannot handle the issues of natural combinatorial complexity.

Tractable two-stage robust optimization (RO) technique in [4] designed for highly tractable and robust to uncertainty. But, the method has no equal tractability and reduces the quality of results. Adaptive Random Testing with Combinatorial Input Domain (ART–CID) in [13] planned for increasing the failure-detection ability of Random Testing (RT) by equally spreading test cases in every input domain.

However, the approach improves the efficiency in combinatorial input domain remains unaddressed. Combinatorial testing in [11] presents pairwise testing for orthogonal arrays and covering arrays that are employed to test every pairs of test settings. Pairwise algorithm with constRaints, Order and Weight (PROW) in [1] considered for executing publicly available web tool called combinational testing in order to analyze the test case in lesser difficulty and higher effectiveness.

Decision to Decision (DD) path graph in [8] achieved to minimize the count of interactions to be tested without loss of fault detection capacity. However, DD path graph is consider for improving the time complexity. A automatically produce test cases in [15] designed for robustness testing of autonomous vehicle control software in closed loop simulation are required. However, the performance of method is to calculate the autonomous vehicle model-based testing scenario is more complicated. Combinatorial Auction for Service Selection (CASS) in [9] improves the quality-aware service selection method for detecting the optimal results and guaranteeing approach efficiently. But, CASS is not investigating the impact of various measures in terms of effectiveness and efficiency.

Cuckoo Search (CS) Algorithm in [7] utilized to design and executes the optimized combinatorial test suite. Though, CS technique has the ability to provide high computational time. Automatic quality signing and verification method in [12] improves the amount of product with minimum expensive during entire resource usage and execution of time. Though, the mutation testing is applied for very big programs that are very hard to detect the better test suite for destroying all the mutants. A combinatorial optimization technique in [5] efficiently recognizing the configuration-aware faults in software system. Though, the performance of combinational technique is planned to minimize the more efficient and effective data structures.

Complete Fault Interaction Location (comFIL) in [3] executes better result compared with known defect location system in CT due to its enhanced effectiveness and accuracy. However, comFIL cannot investigate the effective combinatorial testing process like web service. The PID tuning technique is essential to obtain the interaction of input parameters as a constraint in [14] for constructing the combinatorial set that generates better efficiency and effectiveness. Test case-aware covering array in [17] planned to decrease the quantity of configurations and number of test runs required. But, the cost- and test-case aware covering arrays maintains the general cost technique in which the overall cost of testing are not considered. Combinatorial Interaction Testing (CIT) in [10] is essential since it tests interactions among the several features and parameters create software product. However, CIT technique is determined to reduce the quantity of constraint information.

3. Combinatorial Testing To Improve Software System Quality: Combinatorial testing is processes that are systematically examining the system settings for attaining the quantity of tests. The performance of software systems is very less complicated and necessary to obtain the very large amount of feasible tests cases. Testing is the method of estimating a system or components with the objectives to detect whether it gratify the particular requirements or not. Software testing has the ability to employ the implementation of software component for calculating one or more attributes. Analysis of software testing is performed to generate the information about the number of software result or service under test. In addition, the main objective of combinatorial testing is to improve the test efficiency, speed, reduced expense that achieves the high quality of results in an efficient way. The performance of combinatorial testing to improve software quality is compared against with the three existing methods namely Combinatorial Optimization technique, complete Fault Interaction Location (comFIL) and Pairwise algorithm with constRaints, Order and Weight (PROW) techniques.

3.1 Test case minimization approach using fault detection and combinatorial optimization techniques for configuration-aware structural testing: Test case minimization method is designed to detect the configuration-aware faults in software system and it supports both the combinatorial optimization and sampling technique. Initially, the combinatorial optimization is employed for providing the better test suite with sampling the number of input design. After that, the optimization method produces the test suite that supports the adaptive system by using mutation testing approach. Then the test suite is established to optimize the Cuckoo Search (CS) with combinatorial technique. In addition, the mutation testing is efficiently employed for identifying the various faults to software-under-test and also filters the test cases based on detected faults.

Figure 1 describes the general procedure for cuckoo search strategy. CS is utilized to optimize and investigating an optimal solution by covering the d-tuples list. The CS method is essential for calculating the optimized combinatorial test suites that provides better results in an efficient manner. The strategy measures the effectiveness for detecting the faults in programs by using the functional testing approach. Finally, the method successfully demonstrates that the combinatorial optimization and cuckoo search application to test the software results. Furthermore,

the technique is very essential to achieve the test case prioritization for preceding the test case based on fault density efficiently.

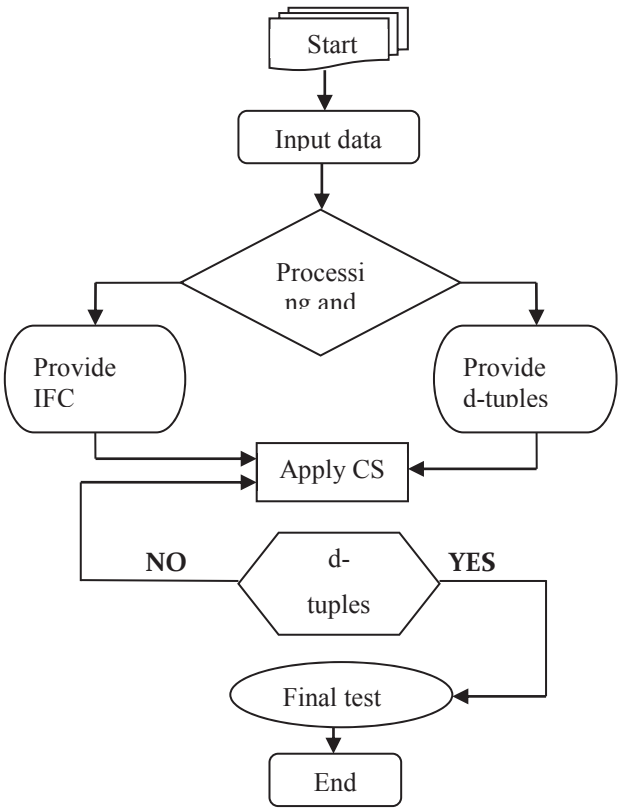

Figure. 1 Flow chart of Cuckoo Search Strategy

3.2 Locating Minimal Fault Interaction in Combinatorial Testing: Combinatorial testing (CT) technique is extensively minimizing the testing cost and enhances the quality of software system. CT method is essential to provide the test suite as input to perform black-box testing has the ability to detect interactions that generate the system's faults. Complete Fault Interaction Location (comFIL) algorithm is presented to find the interactions that cause system's failure and it attain the least set of target interactions during test suite generated with CT. By using comFIL technique, testers investigate and accurately detect the factors which are related to defects of system. Thus the process of creating the software testing and debugging is very simple and more efficient. In addition, ComFIL is more exact in fault localization compared with another technique in terms of CT approach.

The complete Fault Interaction Location (comFIL) system is designed to detect each interaction related to faults. The process of ComFIL consists of two steps which are detected to reduce the fault interactions. After performing the original test cases, the test cases are classified into two sets such as FTS and PTS technique. The former has the test cases that are very essential to trigger faults, while the latter test case does not trigger the faults. Hence, it recognizes the set of interactions covered by FTS but not covered by PTS is termed as candidate Faulty Interaction Set (canFIS). Next, the other test cases are provided to choose the interactions in canFIS method and then successfully achieve the smallest fault interaction set. Finally, the comFIL algorithm is demonstrated to improve the performance while compare with known fault location in combinatorial testing since it provides better effectiveness and accuracy.

3.3 PROW: A Pair wise algorithm with const Raints, Order and Weight: PROW algorithm is mainly designed for pair-wise testing that is essential to allow the tester for managing the constraints and weight. A tool is termed as CTWebadds functionalities that are essential to implement the Pairwise algorithm with constRaints, Order and Weight (PROW) algorithm. The PROW is one of the products sampling technique in Soft-ware Product Lines (SPL) through significance feature approach. Then the improvement of SPL is the new model for software systems that builds the set of common functionalities and some variable functionality. Pairwise testing is employed to achieve the product sampling to test in SPL method by using the features as pairwise parameters. Furthermore, the constraint handling becomes more essential and is presented for evaluating the complexity and efficiency.

The main objective of PROW algorithm is essential to provide the pairwise test suites operate with domain semantics and knowledge. Initially, the algorithm creates feasible for avoiding every pairs among the parameter values are meaning-less. Next, the algorithm has the ability to include those pairs that needs more frequent testing is called as "pair weight". Finally, the ordering test suite is employed to perform test planning, because the essential test cases are required to test previously. Also, it is mainly useful in the regression testing, while the testing time is not enough to perform the complete test suite.

Another contribution of PROW algorithm is designed to execute publicly available web tool, under the GNU license is termed as CTWeb. CTWeb tool is very essential to generate the features that enabled to employ PROW efficiently and also product sampling in SPL. Some of the functionalities are successfully produce the CTWeb for enhancing the PROW methods. At first, product sampling enables the feature model and their relationships to attain the parameters to execute PROW algorithm. CTWeb method is enabled to upload a file, where the tester specifies the parameters, its values and the constraints and associated weight between pairs by using combinatorial testing. In addition, the PROW

technique has the ability to attain the test cases for combinatorial testing approach. Thus the PROW algorithm is efficiently provides the test cases in less time and prevents the inclusion of meaningless pairs. The experimental evaluation using combinatorial testing to improve software system quality is conducted on various factors such as Time complexity, Test execution efficiency, Software Fault Prevention Level.

4. Comparison Of Combinatorial Testing Using Different Techniques And Suggestions: In order to compare the combinatorial testing using different techniques, number of test cases is taken to perform this experiment. Various parameters are used for combinatorial testing for enhancing the quality of software system techniques.

4.1 Time Complexity: Time complexity is defined as the amount of time taken by the process is required to solve the problem while performing the entire test cases. Time complexity is measured in terms of milliseconds (ms) and mathematically formulated as below,

Time complexity(TC)=n*Time(No.of Test cases)

Where, 'n' represents the amount of time taken. When the time complexity is lower, the method is said to be more efficient.

Table 4.1 Tabulation of Time Complexity

Tests Cases	Time Complexity (ms)		
	Combinatorial Optimization Method	comFIL	PROW
10	61	69	55
20	63	72	57
30	64	75	58
40	68	77	62
50	72	79	65
60	74	81	68
70	76	84	71

Table 4.1 describes the time complexity versus different number of tests cases in the range of 10 to 70. The time complexity comparison takes place on existing Combinatorial Optimization technique, complete Fault Interaction Location (comFIL) and Pairwise algorithm with constRaints, Order and Weight (PROW) methods.

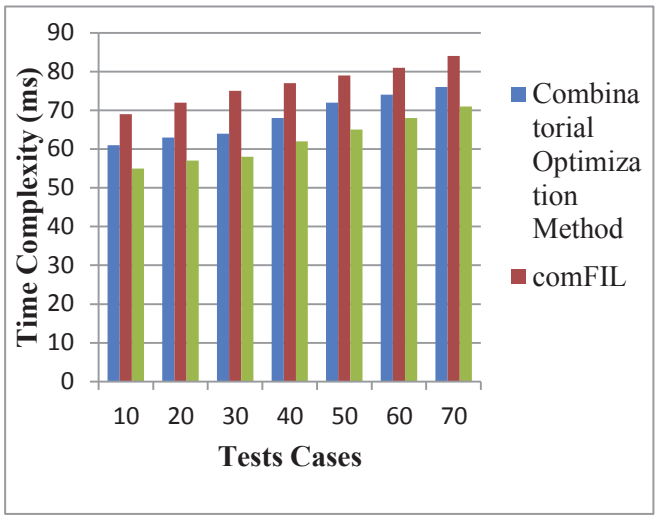

Figure 4.1 Measurement of Time Complexity

Figure 4.1 measures the time complexity of existing techniques. Time complexity of Pairwise algorithm with constRaints, Order and Weight (PROW) technique is comparatively lesser than that of Combinatorial Optimization technique and complete Fault Interaction Location (comFIL) approach. Research in Pairwise algorithm with constRaints, Order and Weight (PROW) method consumes 23% lesser complexity than complete Fault Interaction Location (comFIL) technique and 10% lesser complexity than Combinatorial Optimization technique.

4.2 Test Execution Efficiency: Test execution efficiency (TEE) is defined as the ratio of exactly executing the test cases to the total number of test cases. Test execution efficiency is measured in terms of percentage (%) and mathematically formulated as below,

$$TEE = \frac{Number\ of\ exactly\ executing\ the\ test\ cases}{Total\ number\ of\ test\ cases}$$

When higher the test execution efficiency, the method is said to be more efficient.

Table 4.2 Tabulation of Test Execution Efficiency

Tests Cases	Test Execution Efficiency (%)		
	Combinatorial Optimization Method	comFIL	PROW
10	68	55	62
20	74	58	65
30	77	62	68
40	80	64	70
50	82	67	72
60	84	70	75
70	86	74	78

Table 4.2 describes the test execution efficiency versus different number of tests cases in the range of 10 to 70. The test execution efficiency comparison takes place on existing Combinatorial Optimization technique, complete Fault Interaction Location (comFIL) and Pairwise algorithm with constRaints, Order and Weight (PROW) methods.

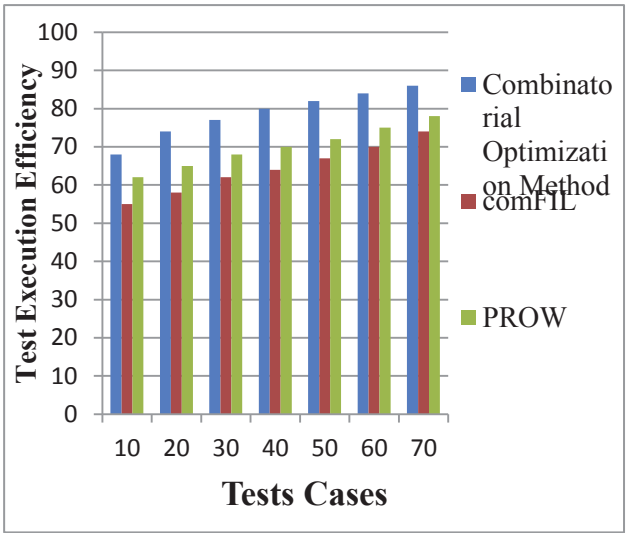

Figure 4.2 Measurement of Test Execution Efficiency

Figure 4.2 measures the test execution efficiency of existing techniques. Test execution efficiency of Combinatorial Optimization technique is comparatively higher than that of complete Fault Interaction Location (comFIL) and Pairwise algorithm with constRaints, Order and Weight (PROW) approach. Research in Combinatorial Optimization technique has 11% higher efficiency than Pairwise algorithm with constRaints, Order and Weight (PROW) technique and 18% higher efficiency than complete Fault Interaction Location (comFIL) technique.

4.3 Software Fault Prevention Level: The software fault prevention level is measured for preventing the faults occurrence during the software under test cases. Software fault prevention level is measured in terms of percentage (%). When higher the software fault prevention level, the method is said to be more efficient.

Table 4.3 Tabulation of Software Fault Prevention Level

Tests Cases	Software Fault Prevention % level		
	Combinatorial Optimization Method	comFIL	PROW
10	70	90	80
20	75	93	82
30	78	95	85
40	80	98	88
50	82	100	90
60	83	102	93
70	84	105	95

Table 4.3 describes the Software Fault Prevention Level versus different number of tests cases in the range of 10 to 70. The Software Fault Prevention Level comparison takes place on existing Combinatorial Optimization technique, complete Fault Interaction Location (comFIL) and Pairwise algorithm with constRaints, Order and Weight (PROW) methods.

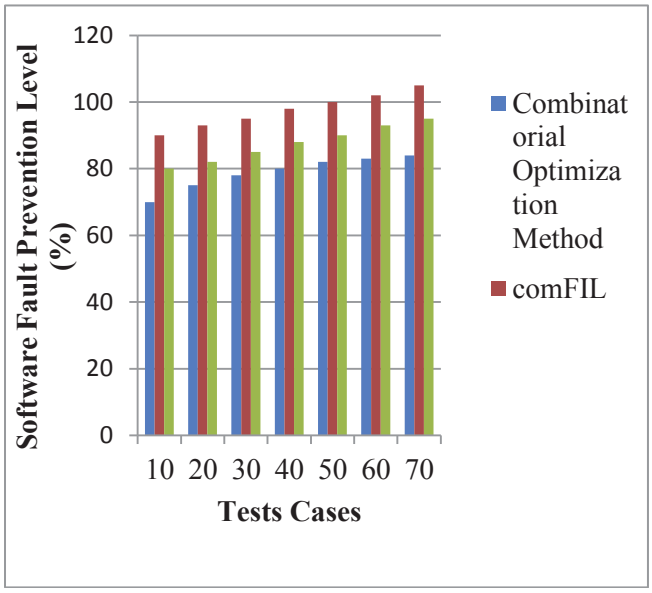

Figure 4.3 Measurement of Software Fault Prevention Level

Figure 4.3 measures the Software Fault Prevention Level of existing techniques. Software Fault Prevention Level of complete Fault Interaction Location (comFIL) technique is comparatively higher than that of Combinatorial Optimization technique and Pairwise algorithm with constRaints, Order and Weight (PROW) approach. Research in complete Fault Interaction Location (comFIL) method has 19% higher software fault prevention than Combinatorial Optimization technique and 10% higher software fault prevention than Pairwise algorithm with constRaints, Order and Weight (PROW) technique.

5. Discussion And Limitation Of Combinatorial Testing To Improve Software Quality Using Different Techniques: Combinatorial Optimization technique is reduced to detect the modern research in mobile application and attributes. The performance of combinational method is planned to minimize the more efficient and effective data structures with better efficient. Developing the approach is not assist with various combination

degrees, seeding and constraints of combination. Thus the method contains low-quality software products that may cause loss of profits or even loss of time.

Complete Fault Interaction Location (comFIL) technique is not essential to investigate the effective combinatorial testing process like web service. The comFIL is not determined to develop the interactions being tested for minimizing the quantity of additional test cases. In Pairwise algorithm with constRaints, Order and Weight (PROW), the testers are not capable to implement the more essential test cases. The definition of rules is to determine the impossible combinations between features remains unaddressed. The time scheduled for testing tasks is not sufficient to operate all the test cases. Therefore, the process of PROW has the ability to provide very high cost and time consuming.

5.1 Future Direction: The future direction of combinatorial testing can be used to decrease the time complexity and cost during the interactions count. In addition, the software system is very easier to find the faults by minimizing the test cases and therefore improves the quality of software product.

6. **Conclusion:** The comparison of different techniques for combinatorial testing to improve software system quality is carried out. The time scheduled for testing tasks is not sufficient to operate all the test cases during PROW system. The PROW technique is essential to improve the expenses and time consuming efficiently. Combinational method contains more proficient and valuable data structures are designed to be unaddressed. Thus the process of comFIL is reduced to identify the significant software quality and combinatorial testing strategy. The comFIL is not evaluated to develop the interactions being tested for decreasing the amount of additional test cases. Finally, from the result, the research work can finding the faults by reducing the test cases and enhances the quality of software system during the interactions count. Furthermore, the technique is achieved to improve the test execution efficiency, software fault prevention level and minimized to consume time complication by using combinatorial testing with better efficient.

References:

1. Beatriz Pérez Lamanchaa,, Macario Polob and Mario Piattini, “PROW: A Pairwise algorithm with constRaints, Order and Weight”, ELSEVIER: The Journal of Systems and Software, Year: Jan 2015, Volume: 99, Pages: 1-9.
2. J. Jeba Jesintha, K. Ezhilarasi Hilda, Sub Divided Uniform Shell Bow Graphs Are one Modulo; Mathematical Sciences international Research Journal ISSN 2278 – 8697 Vol 3 Issue 2 (2014), Pg 645-647
3. Zhaojie Xue, Canrong Zhang, Peng Yang and Lixin Miao, “A Combinatorial Benders’ Cuts Algorithm for the Local Container Drayage Problem”, Mathematical Problem in Engineering, Year: Feb 2015, Volume: 7, Pages: 1-7.
4. Wei Zheng, Xiaoxue Wu, Desheng Hu and Qihai Zhu, “Locating Minimal Fault Interaction in Combinatorial Testing”, Advances in Software Engineering, Year: April 2016, Pages: 10.
5. M.Geethalakshmi, A.Praveen Prakash, A Study on Problems Faced By It Professionals; Mathematical Sciences international Research Journal ISSN 2278 – 8697 Vol 3 Issue 2 (2014), Pg 639-644
6. Bo Zhang, Tao Yao, Terry L. Friesz and Yuqi Sun, “A tractable two-stage robust winner determination model for truckload service procurement via combinatorial auctions”, ELSEIVER: Transportation Research Part B, Year: Aug 2015, Volume: 78, Pages: 16-31.
7. Bestoun S. Ahmed, “Test case minimization approach using fault detection and combinatorial optimization techniques for configuration-aware structural testing”, ELSEVIER: Engineering Science and Technology, an International Journal Year: June 2016, Volume: 19, Issue: 2, Pages: 737-753.
8. Dr. P. Thirunavukarasu,R. Suresh, Soft Complex Fuzzy Set; Mathematical Sciences International Research Journal ISSN 2278 – 8697 Vol 3 Issue 1 (2014), Pg 72-76
9. Sangeeta Sabharwal , Priti Bansal, Nitish Mittal and Shreya Malik, “Construction of Mixed Covering Arrays for Pair-wise Testing Using Probabilistic Approach in Genetic Algorithm”, Arabian Journal For Science and Engineering, Year: Aug 2016, Volume: 41, Issue: 8, Pages: 2821-2835.
10. Bestoun S. Ahmed, Taib Sh. Abdulsamad and Moayad Y. Potrus, “Achievement of Minimized Combinatorial Test Suite for Configuration-Aware Software Functional Testing Using the Cuckoo Search Algorithm”, Information and Software Technology, Year: Oct 2015, Volume: 66, Pages: 13-29.
11. Sangeeta Sabharwal and Manuj Aggarwal, “A novel approach for deriving interactions for combinatorial testing”, Engineering Science and Technology, an International Journal, Year: June 2016.
12. S.Sudha , M.Satheeshkumar, on Lexicographic Product of Paths and Cycles; Mathematical

Sciences International Research Journal ISSN 2278 – 8697 Vol 3 Issue 1 (2014), Pg 77-83
13. Qiang He, Jun Yan, Hai Jin and Yun Yang, "Quality-Aware Service Selection for Service-Based Systems Based on Iterative Multi-Attribute Combinatorial Auction", IEEE Transactions on Software Engineering, Year: Jan 2014, Volume: 40, Issue: 2, Pages: 192-215.
14. Justyna Petke, Myra B. Cohen, Mark Harman and Shin Yoo, "Practical Combinatorial Interaction Testing: Empirical Findings on Efficiency and Early Fault Detection", IEEE Transactions on Software Engineering, Year: Sept 2015, Volume: 41, Issue: 9, Pages: 901-924.
15. Jagdeep Kaur, Methods to Find the Rank and Multiplicative inverse of Fully Fuzzy Matrices; Mathematical Sciences international Research Journal ISSN 2278 – 8697 Vol 4 Issue 1 (2015), Pg 118-122
16. Raghu N. Kacker, D. Richard Kuhn, Yu Lei and James F. Lawrence, "Combinatorial testing for software: An adaptation of design of experiments", ELSEVIER: Measurement, Year: Nov 2013, Volume: 46, Issue: 9, Pages: 3745-3752.
17. Mohammed I. Younis and Kamal Z. Zamli, "A Strategy for Automatic Quality Signing and Verification Processes for Hardware and Software Testing", Advances in Software Engineering, Year: 2010, Volume: 5.
18. Rubing Huang, Jinfu Chen and Yansheng Lu, "Adaptive Random Testing with Combinatorial Input Domain", International Conference on Automated Software Engineering, Year: Nov 2013, Volume: 46, Issue: 9, Pages: 3745-3752
19. V. Shyam Prasad, Dr. P. Kousalya, Consistency of Pair-Wise Comparison Matrix in Analytic Hierarchy Process – An Illustration; Mathematical Sciences international Research Journal ISSN 2278 – 8697 Vol 4 Issue 1 (2015), Pg 137-142
20. Mouayad A. Sahib, Bestoun S. Ahmed, andMoayad Y. Potrus, "Application of Combinatorial Interaction Design for DC Servomotor PID Controller Tuning", Journal of Control Science and Engineering, Year: Jan 2014, Volume 2014.
21. KevinM. Betts and Mikel D. Petty, "Automated Search-Based Robustness Testing for Autonomous Vehicle Software", Modelling and Simulation in Engineering, Year: July 2016, Volume:2016.
22. Bestoun S. Ahmed, Mouayad A. Sahib and Moayad Y. Potrus, "Generating combinatorial test cases using Simplified Swarm Optimization (SSO) algorithm for automated GUI functional testing", Engineering Science
23. and Technology, an International Journal, Year: 2014, Volume: 17, Pages: 218-226.
24. Cemal Yilmaz, "Test Case-Aware Combinatorial Interaction Testing", IEEE Transactions on Software Engineering, Year: May 2013, Volume: 39, Issue: 5, Pages: 684-706.
25. Vidya M Shettar, Shanmukhappa A Angadi, A Genetic Algorithm For Graph Matching Using Node Characteristics; Mathematical Sciences International Research Journal : ISSN 2278-8697Volume 4 Issue 2 (2015), Pg 347-351
26. Zhiqiang Zhanga, Jun Yan, Yong Zhao and Jian Zhang, "Generating combinatorial test suite using combinatorial optimization", The Journal of Systems and Software, Year: 2014, Volume: 98, Pages: 191-207.

M. Bharathi
Assistant Professor, Department of Computer Science, Periyar University College ,
Pennagaram, Dharmapuri.(TN).
Dr. V. Sangeetha
Assistant Professor, Department of Computer Science, Periyar University College.
Pappireddipatty Dharmapuri (TN).