Journal of Theoretical and Applied Information Technology
15th September 2019. Vol.97. No 17

ISSN: 1992-8645 www.jatit.org E-ISSN: 1817-3195

HEURISTIC BASED REGRESSION TEST CASE PRIORITIZATION ALGORITHM WITH ANALYSIS FOR TEST COST REDUCTION AND OPTIMIZED COST PATH GENERATION

[1]DR. PRITI, [2]KAVITA

[1]Assistant Professor, Computer Science and Application Department, MDU, Rohtak, Haryana, India

[2]Research Scholar, Computer Science and Application Department, MDU, Rohtak, Haryana, India

E-mail: [1]pritish80@yahoo.co.in, [2]kavita.dahiya78@gmail.com

ABSTRACT

Regression testing applies the selective test method on software system to identify the faults in modified software system in the maintenance phase. The observation criteria, test case priority and sequence of test conductance affect the testing efforts, cost and efficiency. In this paper, a level prioritized and weight processed dynamic programming method to optimize the regression test sequence. In this proposed model, the work is divided in three integrated stages. In first stage, the graph based dependency and branch analysis is done. At this stage, the prioritization of modules and module-segments is done under graph degree and dependency evaluation. In second stage, various aspects of fault and code change are analyzed collectively to assign the priority code segments. The module weights are also integrated with priority vector to generate the collective weights for each stage. In the final stage, the dynamic programming approach with back aggregative evaluation is conducted to identify the optimized test sequence. The comparative evaluation is conducted against fault based and dependency based evaluation methods applied using greedy and dynamic programming approaches. The analysis results verified that the proposed model reduced the cost and number of testcases required to perform the regression testing for software projects. The overall performance of regression testing is improved by this proposed approach.

Keywords: *Regression Testing, Prioritization, Graph based, Dynamic Programming*

1. INTRODUCTION

Today, the software systems are updated regularly because of the frequent change in client requirements. These changes can be in terms of platform, appearance or the inclusion of some utility. These requirements and changes are identified after the delivery of earlier software system to client end. As the client start getting familiar to the software system, he gets more requirements to gain the better reachability, usage, security and performance for software system or for specific utility. The lesser time and bug free development is the main requirement for modification and updation of these extensions to the software system. The regression testing is the approach used to test the selective modules so that the new bug free version can be delivered to the client. The regression testing process includes the selection of the most effective modules from software system that are required to test. The module selection, dependent module selection, test case selection and the sequence of test case implementation are the major integrated tasks of regression testing. The integrated functions of regression testing are shown in figure 1.

Figure 1 has provided the detailed view of all the integrated sub stages or types of regression testing methods. Each of these methods requires a complete scientific process to optimize the delivery of modified software system. The test case selection is the prior subtype of regression testing that deals in the selection of most appropriate test cases which are required to test the new software system. The test case selection is based on parametric evaluation so that the bug in new modules and integrated software system will be identified. The module dependency, fault occurrence, bug type occurrence etc. are the common parameters used to decide the test cases for regression testing. The test case prioritization method is defined for identification to decide the sequence and execution of the test cases. The prioritization can be applied to process the

ISSN: 1992-8645 www.jatit.org E-ISSN: 1817-3195

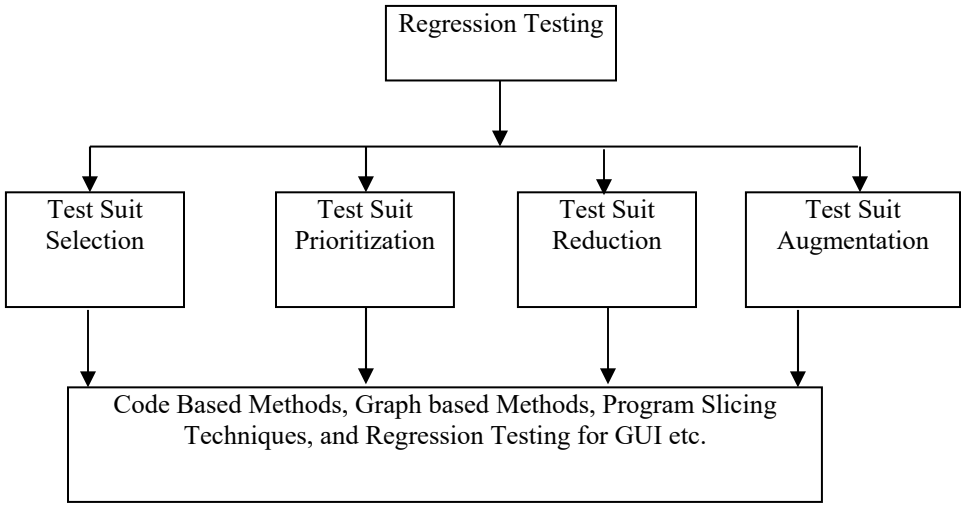

Figure 1: Regression Testing

regression testing in optimum way. The prioritization methods were processed by the researchers using various parameters including statement coverage, fault detection capability, execution cost, fault frequency and module coverage capability. The effective test execution sequence can reduce the software maintenance cost and time. Another scope of regression testing is to reduce the possible test suite. This testing model able to identify the non required test suits which are not relevant to newly included modules and its dependent modules. The coverage, UML analysis and control flow graph based analysis can be applied to identify the most eligible test cases and suites. The correct identification of most suitable test suite can reduce the efforts to test the modified software system. The test case augmentation is another technique of regression testing used to perform code level check. The contributing code elements and code segments can be identified by using this technique. The significance, dependency, coverage, requirements and behaviour of these code segments can be analyzed in this technique to filter the test cases and to improve the reliability of regression testing.

1.1 Objectives

The paper defined a dynamic programming based method for optimizing the effectiveness of test sequence generation for regression testing. The main objectives of this paper are:

- ✓ The graph adaptive dynamic programming model is presented for sequence optimization for regression testing.
- ✓ Algorithm and measures are defined for assigning the priorities to test cases.
- ✓ The analytical observations are conducted on five real time projects.
- ✓ The cost based evaluation is conducted against the greedy and dynamic programming based methods.

In this paper, graph based dynamic programming method is applied in this work to optimize the performance and effectiveness of regression testing. The proposed model fault features, module dependency features and graph coverage measures collectively to optimize the path sequence. In section 1, the basic requirement and functioning of regression testing is defined. Different elements and types of regression testing are also defined with specification of its scope. In section 2, the literature work done by the researchers is discussed. In section 3, the proposed graph based regression testing method is presented. In section IV, the cost based evaluation results are provided.

2. RELATED WORK

Regression testing is integrated as the essential stage in the software projects to reduce the cost and time to analyze the recent changes. Researchers have defined various methods to generate the test cases and test sequence for improving the regression testing. Researchers have used the prioritization measures and optimization methods to improve the scope and behaviour of regression testing. In this section, the contribution

ISSN: 1992-8645 www.jatit.org E-ISSN: 1817-3195

of the earlier researchers is discussed in improvement of regression test methods.

Indumathi et al.[1] has used the genetic method to prioritize the test cases and to improve the efficiency and reliability of software test activity. Author reduced the overall testing cost with maximum coverage and fault rate consideration. The proposed prioritization based test sequence reduced the fault loss and execution time. Vescan et al.[2] used the test case prioritization as the consistent part to reorder the test cases. The requirement dependencies were considered by the author as integrated approach to improve the ability of fault detection with lesser executions cost. Morozov et al.[3] has used the updated block error and propagation error based method to prioritize the test cases. Author used the Markov based stochastic propagation analysis to validate the test cases. Author tested the method by injecting the faults at various level and observed the probability of fault detection by the proposed method. Al-Refai et al.[4] has used the fuzzy logic based model to analyze the UML diagram to identify the degree of uncertainty. The traceable link based probabilistic evaluation was conducted to prioritize the test cases. The code based design model was provided by the author to refine the test sequence. The structural matching and fault detection ability was included to minimize the prioritization cost in regression testing. Suleiman et al.[5] has provided a study on various prioritization techniques for regression testing. The requirement based, model based and coverage based methods available for test case prioritization were discussed by the author. Brahneborg et al.[6] has identified the various challenges exist in regression testing in pragmatic perspective. The structural and theoretical behaviour of the regression test methods were discussed by the author to identify the failure and to integrate it with real environment. The organization specific and the structural behaviour of software system were also discussed by the author.

Strandberg et al.[7] has provided a discussion of prioritization method at system level regression test methods. The regression test cost minimization and prioritization method were explored by the author. The manual and automated methods were discussed in this research for implementing the regression testing. Hafez et al.[8] has used the method to identify the prone error and to identify the error evolved in the system. The frequency based and the criticality based evaluation was discussed by the author to identify the fault detection rate in the system. The link specific evaluation was discussed by the author to improve the detection of fault in the system during regression testing. Alagoz et al.[9] has regulated a stochastic model for identification of failure probability in the system and to identify the failure during regression testing. The mutual differential information and failure probability was identified by the author through validation. The differential information with failure degree evaluation was discussed to reduce the system risk. The mutual differential analysis was presented to improve the system reliability and to quantify the failure probability in the system. The parameter level estimation was used to identify the probability of fault occurrence in the system. Ammar et al.[10] has provided an enhanced weighted method to prioritize the test suite. The random ranking and performance optimization method was suggested by the author to generate the weights. The coverage score, branch score and fault coverage were used by the author in weighted form to identify the cost of each regression test. The method has provided the maximum coverage for different prioritization criteria to improve the performance of regression testing.

Haider et al.[11] has used the neuro-fuzzy system to optimize the generation of regression test suite. The coverage cost specific evaluation was suggested by the author to reduce the cost of optimization process. The method was applied on infrastructure repository to validate the software testing. Wang et al.[12] has used the history based dynamic test case prioritization method to optimize the regression testing for industrial system. The study was provided to measure the average percentage of faults and the fault detection rate. The requirement properties and fault detection at significant factor was analyzed by the author to adjust the priorities dynamically. The history driven evaluation had improved the estimation of candidate contribution and improved the reliability of test case prioritization. Strandberg et al.[13] has proposed the automated tool called suite-builder to observe the test cases. Various features of regression test cases and the prioritization criteria were analyzed by this model. These associated factors were analyzed under fault criticality and bug frequency factors. The experimentation was conducted by the author to detect the faults and to generate the significant regression testing results. Dini et al.[14] has designed an automated test case generation system using feedback directed random technique and search based technique. The fine-grained method was defined by the author to track the dependencies and to optimize the test sequence.

ISSN: 1992-8645 www.jatit.org E-ISSN: 1817-3195

The test selection and grouping method was also discussed in this research. Priyanka et al.[15] has proposed the Revised AHP (Analytical Hierarchy Process) method to prioritize the test cases. The method classifies the test cases based on multiple criteria and later generates the pairs on the available criteria. The substitute method was provided by the author to generate the relative weights for assigning the priorities of regression test cases.

Dhareula et al.[16] has provided an exploratory study on regression test selection under attribute reusability evaluation. The requirement based method was provided by the author to test the granularity of cost and to improve the test case selection. The fine granularity and coarse granularity method was evaluated by the author

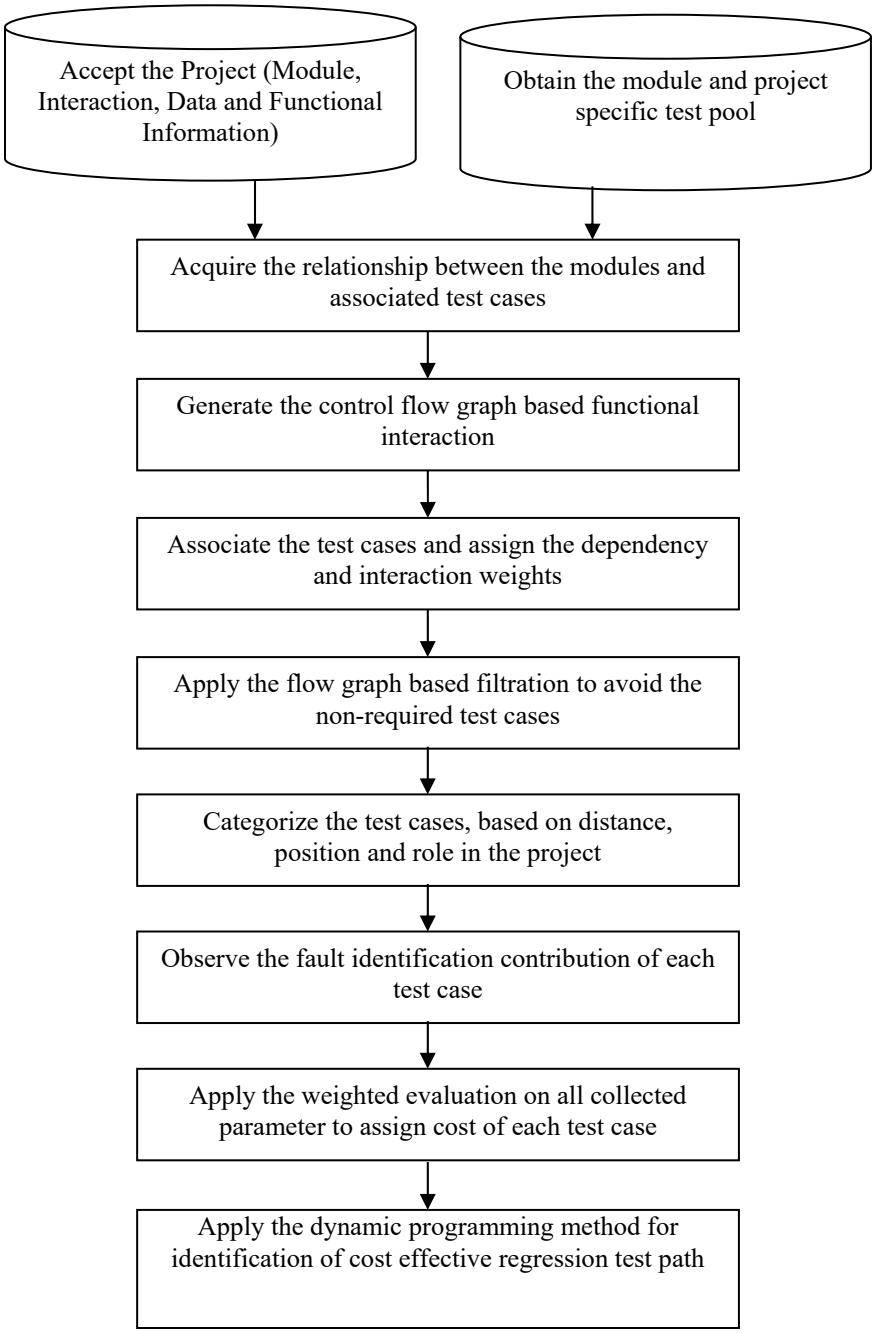

Figure 2: Graph Analysis based Dynamic Programming approach for Regression Test Sequence Optimization

ISSN: 1992-8645 www.jatit.org E-ISSN: 1817-3195

under cost and level parameters. Srisura et al.[17] has provided novel false test case identification method for improving the test case selection behaviour. The systematic study was also conducted to ensure the validity of test case selection. The test size and selection criteria behaviour was also discussed in this research to optimize the test sequence. Akimoto et al.[18] has used the control flow graph based method for optimizing the test case selection for regression testing. The complexity evaluation and accuracy evaluation was also observed by the author to identify the dependency between the test cases. The metrics based complexity evaluation method was discussed in this research. Kumar et al.[19] has provided the modified-ACO method to maintain the diversity and to optimize the test path. The huge search space was processed by the author to identify the maximum fault in minimum time. The convergence behaviour was analyzed within ACO method to reduce the cost and effort of regression testing. Dahiya et al.[20] has provided the novel approach to analyze the classes, sequences and activity diagrams. The tool was designed by the author under the consideration of reusability and semantic behaviour of test cases. The study was provided to identify different kind of statement and relative prioritization criteria and methods. Kandhil et al.[21] has presented the automated agile regression based testing approach to expose the similarity issues. The prioritization method was defined by the author to generate the weighted agile parameters. Gao et al.[22] also used the ACO under fault, execution time and fault severity parameters to prioritize the test cases. The composite metrics based method provided the effective reordering of regression test cases. Ekelund et al.[23] has used the history based evaluation to optimize the test sequence. The recommendation and correlation evaluation method was defined by the author to improve the correctness of regression test sequence. Chauhan et al.[24] has used the model based method for effective selection of test cases for regression testing. The data dependency in graph representation was provided by the author to reveal the faults and to optimize the test sequence. El-Din et al.[25] has provided the cost included greedy based test sequence generation method to reduce the cost and effort of regression test optimization. The behaviour and evaluation of regression test method was provided by the author.

From the study we observed that the lot of work is done by the researchers for test case prioritization. The fault based, module based and dependency based methods were defined by the researchers for assigning the priorities to testcases. The researchers also used various swarm based, mathematical and evolutionary methods for optimizing the test sequences. The study identified that the researchers lacks the use of composite prioritization method by considering the multiple parameters collectively. Most of the researchers used the forward analysis approach for generating the effective test sequence. From the study, the main gap identified is the use of some backward processor so that the effective and cost driven test sequence will be obtained. The presented research model is able to cover these research gaps and able to identify the cost effective test sequence for real time projects.

2.1 Significance of Work

In this research, graph based dynamic programming method is defined for optimizing the test sequence for regression testing. The significance of the proposed model over existing methods is given below:

- ✓ In the existing work, most of researchers used the single measure such as fault type, fault count, fault severity or the module dependency. But in this research, we have defined a composite weighted method based on fault features and module dependency measure so that the reliability and effectiveness of work is improved.
- ✓ In the existing work, a forward map and optimization methods were used for identifying the effective test sequence. In this research, the back tracked dynamic programming method is defined for optimizing the test sequence.

3. RESEARCH METHODOLOGY

In this section, the Graph analysis based Weighted Dynamic Programming model is defined to optimize the regression test path sequence. The proposed model shown in figure 2 has acquire the software project features including the module information, functional connectivity between modules and the data sharing within modules. Once the project information is acquired, the next task is to generate the test cases associated to these modules, functional descriptions and data members. By performing this analysis a vast pool of available modules and associated test cases is obtained. Now, the relationship between these modules and available test cases is done.

ISSN: 1992-8645 www.jatit.org E-ISSN: 1817-3195

In this association stage, the contribution, position and the usage of each test case is obtained. Now the functional flow graph is generated based on the functional flow of software project to generate the graph based view of available test cases. This flow is defined with specification of initial, intermediate and modules and functional codes. Once such relationship is obtained, the test cases are observed to generate the functional dependencies and interaction between the modules and test cases. The graph flow analysis based filtration stage is applied to avoid the non-required testcases. The categorization of test cases is also done based on the position, distance and the type of module it is associated. The usage and the scope of the testcases are analyzed to identify the category of test cases. Another evaluation is done to identify the fault or the problem addressed by the testcase. Based on these all parameters the cost of the testcases is obtained. Once the control flow graph with relative cost based testcases specification is obtained, the final stage is to apply the dynamic programming method to generate the optimized test path so that the overall cost of regression testing will be reduced. The process of control flow graph based testcase processing is provided in algorithm I. In this algorithm, the algorithmic process of weight generation and cost evaluation is provided.

Algorithm 1: Proposed Regression Testing Algorithm

```
RegressionTest (TestCases)
/*Testcases is the sequence of graph transformed test cases for the software project or program code*/
{
1.      For i=1 to Testcases.cases
        do
2.      Testcase=Testcases(i)
        /*Get the test case from the graph framed test sequence*/
3.      level=GetLevel(Testcase)
        conn1=GetPrevConn(Testcase)
        conn2=GetNextConn(Testcase)
        /*Get the dependency level for identification of level of test cases*/
4.      Testcases.Score(i)=conn1*w1+conn2*w2+(Maxlevel-level)*w3
        /*Identify the score to classify the testcases*/
        End
5.      For i=1 to Testcases.cases
        do
6.      faultP=GetFaultParameters(Testcases(i))
        /*Get the fault level evaluation*/
7.      priority(i)=faultP.faultCriticality*w1+faultP.faultFreq*w2+TestCase(i).Criticality*w3
        /*Identify the test case priority based on different fault criticality aspects*/
8.      priorityS(i)=Testcases(i).score(i)*priority(i)
        /*Apply the priority of the test cases based on the level score of test cases*/
        End
9.      Tcost=0
        /*Initial test cost is 0*/
10.     For i=levels.N to 1
        /*Apply dynamic programming in reverse aggregative evaluation measure for test sequence generation*/
        do
11.     testcases=GetTestcases(levels(i))
        /*Get the testcases for the particular level*/
12.     testC=GetOptimized(testcases,priorityS)
        /*Get the optimized test case for the particular level*/
13.     Tcost=Tcost+GetCost(testC)
        /*Get the total cost of testcase of that level*/
        End
14.     Return Tcost

}
```

Algorithm I has provided the method to process the testcases defined as the control flow graph. As a test case is processed, its dependency and the level based distance analysis is performed. The dependency and interaction based analysis is done to generate the score of testcases. The fault, criticality based analysis is performed for each of the testcases to generate the priority of the test cases. Once the levels, weights and priority of testcases are obtained, the dynamic programming method is applied on control flow graph to identify the optimum test sequence. This dynamic programming method is able to optimize the regression testing.

Dynamic programming approach is applied in this work to analyze the software project or code for effective test path. To formulate the system under mathematical rules, it is required to analyze the system technically according to the algorithmic approach. In this work, the complete

ISSN: 1992-8645 www.jatit.org E-ISSN: 1817-3195

software system is divided in smaller segments and each segment is defined by a level. The level based analysis applied in this work is given in equation (1)

$$r(t) = [r_1(t), r_2(t), ..., r_N(t)] \quad (1)$$

Here r1,r2…rn are representing the levels in the software system and t is the index of these levels. r(t) represents the aggregative result representing the complete system dimension. The behavior of the algorithm considers the initial stage and process with subsequent stages while handling the pre-determined conditions and constraints. Let the conditions defined in this section includes p0=p as the initial stage and , $p_{n+1} = W(p_n)$ with the weighted vector. The moment observation along with process definition is described at each stage. The generated set of vector is given by (p0,p1....pn). The iterative process relative to the weighted solution is shown in equation (2)

$$p_n = W^n(p) \quad (2)$$

Here the weight analysis is applied with individual stage analysis. Each stage is divided in number of test cases. Here n is the number of test cases in each analysis is applied to obtain the identification of cost effective test case.
The method also includes the analysis at each stage with recursive inclusion of process to apply a cyclic analysis. The function in this method is described by $\sum_{i=0}^{N} G(p_i)$ which provide the taransformed relation for the relational analysis. The recursive relation is here presented by the equation (3)

$$f_N(p) = \sum_{i=0}^{N} G(p_i), \qquad N = 0,1,2,... \quad (3)$$

The work is here presented as the functional observation so that the recursive tranformation is shown in equation (4)

$$f_N(p) = G(p) + f_{N-1}(p_1) = G(p) + f_{N-1}[W(p)], N = 1,2,... \quad (4)$$

The method includes the series of vector defined at each stage and the relation observation is extended with parameter based specification so that the transformed vector will be obtained from the system. The sequence selection and process measurement is here defined in a relational form so that the extension to the network is applied for real transformation to the vector with weighted specification in probabilistic form given by $p_n = W^n(p)$. The method is here defined as the influenceing model with two parameters p and q. Here p represnets the fault specific analysis and q is representing the interaction level observation.

4. RESULTS

The graph analysis based dynamic programming method is defined in this work to optimize the regression testing. The proposed model is implemented in matlab environment. The input taken for the evaluation is in terms of control graph. The control graph is generated by evaluating the available software project. To generate the analytical results of regression test optimization, the evaluation is performed on five software projects. The projects are processed at the earlier stage to acquire the number of modules, functional interactions and data interactions. After evaluating the functional and data features of software projects, the associated test cases are identified. The project name and associated test cases are shown in table 1. These test cases are associated to the particular modules and functional behaviour of projects.

Table 1 : Test Cases in different Projects

Projects	Total Test Cases
Library System	79
Hospital Management	116
Inventory	62
Exam Processing	86
Hotel Management	74

ISSN: 1992-8645 www.jatit.org E-ISSN: 1817-3195

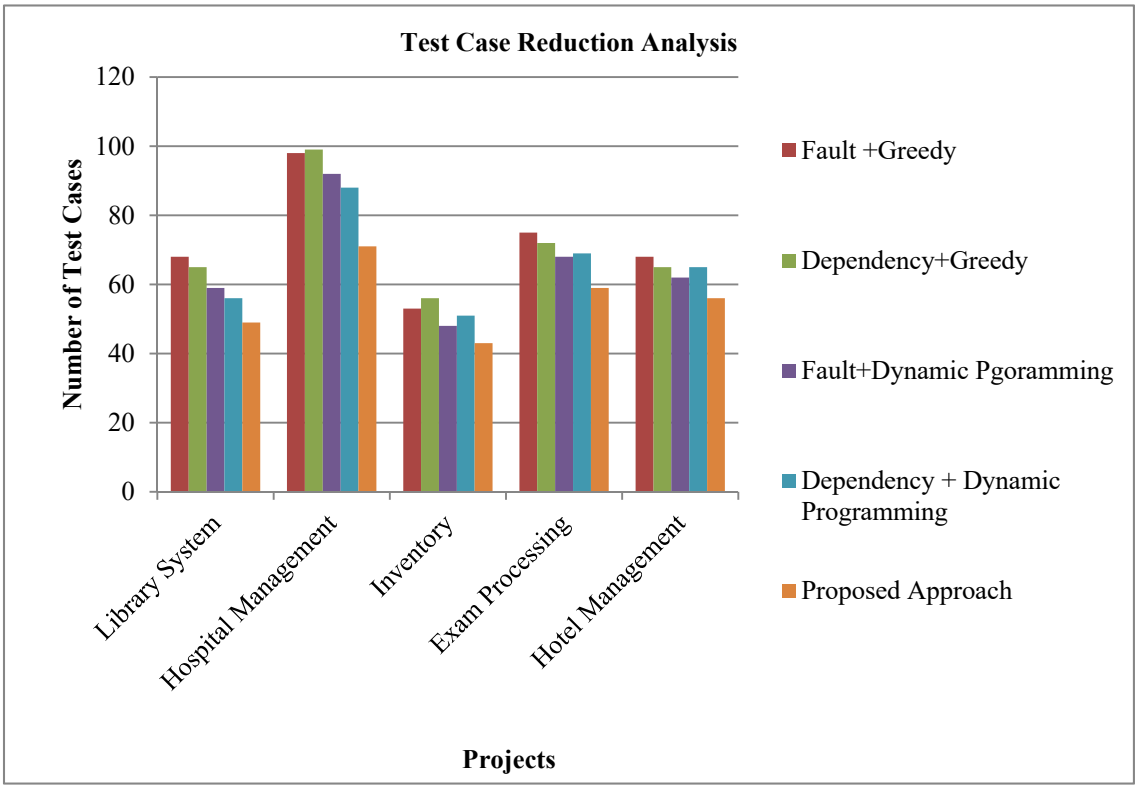

Figure 3: Number of Reduced Test Cases Processed

The project test cases are framed as the control flow graph with specification of various associated features including the dependency, distance, position, interactions, criticality level and the type of faults addressed by the testcases. In the final stage, the dynamic programming method is applied to generate the optimized test sequence. The method is defined to reduce the number of processing testcases as well as to reduce the cost of the test sequence generation. The comparative evaluation of this work is done against the greedy based test path generation approach. The existing greedy based and dynamic programming based methods are evaluation with specification of fault and dependency based weights for evaluation of testcase cost. The numbers of testcases in earlier stage are provided in table 1. After applying the regression testing, some of the test cases are avoided. The removal of such testcases from pool is called test case reduction. The comparative evaluation results based on this parameter are provided in figure 3.

Table 2 : Comparative Analysis of Testcase Reduction

Total Test Cases	Fault +Greedy	Dependency + Greedy	Fault+ Dynamic Programming	Dependency + Dynamic Programming	Proposed Approach
Library System	13.92405	17.72152	25.31646	29.11392	37.97468
Hospital Management	15.51724	14.65517	20.68966	24.13793	38.7931
Inventory	14.51613	9.677419	22.58065	17.74194	30.64516
Exam Processing	12.7907	16.27907	20.93023	19.76744	31.39535
Hotel Management	8.108108	12.16216	16.21622	12.16216	24.32432

ISSN: 1992-8645 www.jatit.org E-ISSN: 1817-3195

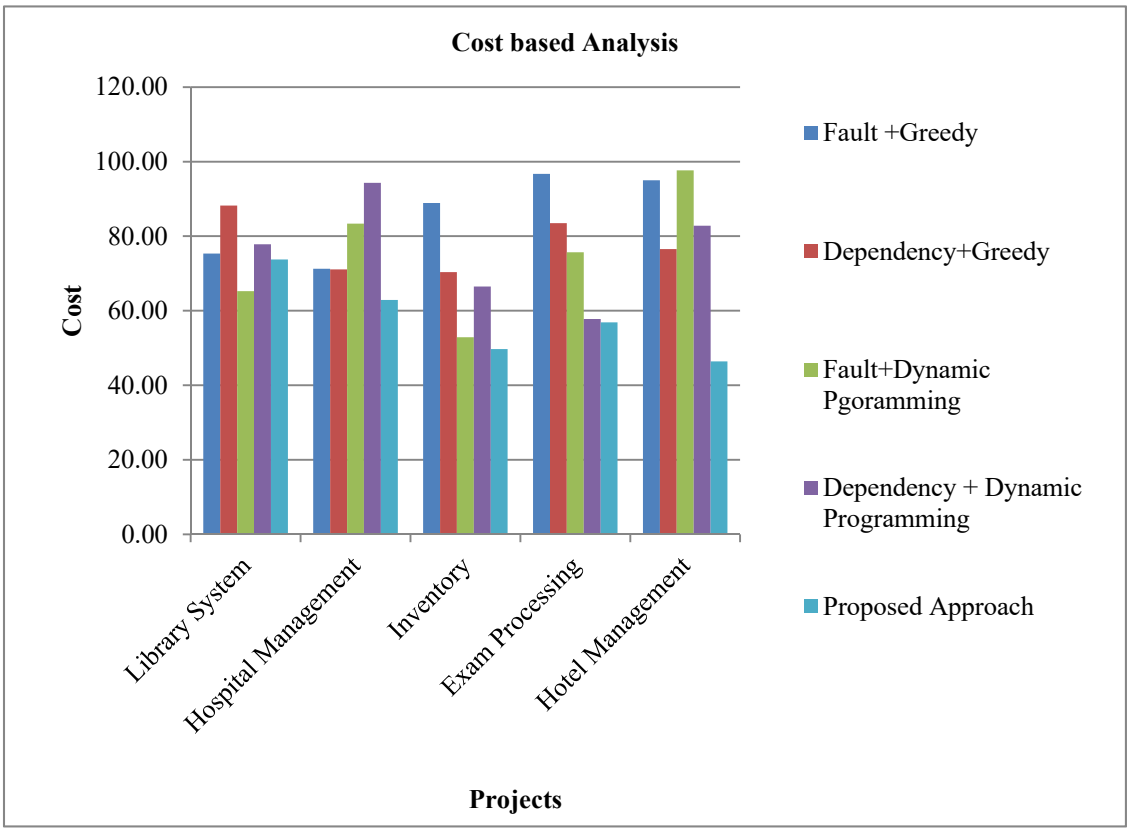

Figure 4 : Cost based Comparative Analysis

The results of figure 3 and table 2 identified that the proposed graph based dynamic programming approach reduced the maximum number of testcases. The maximum test case reduction for each method is achieved for hospital management project and the least testcase reduction is achieved for hotel management project. Another analysis of the proposed method is done in terms of cost evaluation. The cost is defined as the effort to process the particular test case. This cost depends on the test case position, critical, module integration and the fault representation of that particular test case. The cost based analysis results for regression testing optimization are provided in figure 4.

Figure 4 has provided the comparative analysis of the software project for different regression test optimization methods. In this figure, x axis represents the software projects and y axis shows the cost of the test sequence generation. The graph clearly shows that the cost of regression test implementation is minimum by using proposed approach. The results shows that the cost of regression test optimization is higher only for library system project for rest of other projects, the proposed method has generated the results at minimum cost. The results identified that the proposed approach is significantly better to improve the proposed of regression testing.

4. CONCLUSION

In this paper, a two stage evaluation based dynamic programming model is presented to identify the optimized sequence of regression test cases. The model is divided in three stages. In first stage, various aspects of dependency evaluation area applied on test framed graph. The degree and connectivity analysis is performed to identify the score of each test case. In second stage, the test cases are analyzed under different kind of fault issues. The fault criticality, module criticality and fault frequency are analyzed in this stage in weighted from. This fault criticality is also combined with score of each testcase. In the final stage, the level specific evaluation is performed under weighted multiple criteria. In this stage, the dynamic programming model is applied to generate the optimized test sequence. The proposed model is

ISSN: 1992-8645 www.jatit.org E-ISSN: 1817-3195

applied on five real time projects and analysis is performed in terms of cost and fault identification in software projects. The comparative results are generated against the dependency and fault based methods. The comparative results identified that the proposed hybrid weighted model has reduced the cost of regression testing and improved the system reliability.

In this paper, a hybrid optimization model is presented using dynamic programming approach for test sequence optimization. The paper used the fault parameters and dependency as the key criteria for deciding the priorities of test cases. In future, some other parameters at code level, module level and functional level can be used for enhancing the performance of the mode. The work is defined specifically for object based or object oriented languages. In future, the work can be extended broadly for other languages and environments. The present work can be extended in future by using some swarm or evolutionary optimization methods. The ACO (Ant Colony Optimization) or ABC (Artificial Bee Colony Optimization) methods can be applied under the particle behavior integration for reducing the cost of test sequence generation. The genetic based evolutionary optimization approach be applied for optimizing the test sequence.

REFERENCES:

[1] C. P. Indumathi and S. Madhumathi, "Cost aware test suite reduction algorithm for regression testing," 2017 International Conference on Trends in Electronics and Informatics (ICEI), Tirunelveli, India, 2017, pp. 869-874.

[2] A. Vescan, C. Serban, C. Chisalita-Cretu and L. Diosan, "Requirement dependencies-based formal approach for test case prioritization in regression testing," 2017 13th IEEE International Conference on Intelligent Computer Communication and Processing (ICCP), Cluj-Napoca, 2017, pp. 181-188.

[3] A. Morozov, K. Ding, T. Chen and K. Janschek, "Test Suite Prioritization for Efficient Regression Testing of Model-Based Automotive Software," 2017 International Conference on Software Analysis, Testing and Evolution (SATE), Harbin, 2017, pp. 20-29.

[4] M. Al-Refai, W. Cazzola and S. Ghosh, "A Fuzzy Logic Based Approach for Model-Based Regression Test Selection," 2017 ACM/IEEE 20th International Conference on Model Driven Engineering Languages and Systems (MODELS), Austin, TX, 2017, pp. 55-62.

[5] D. Suleiman, M. Alian and A. Hudaib, "A survey on prioritization regression testing test case," 2017 8th International Conference on Information Technology (ICIT), Amman, 2017, pp. 854-862.

[6] D. Brahneborg, W. Afzal and A. Causevic, "A Pragmatic Perspective on Regression Testing Challenges," 2017 IEEE International Conference on Software Quality, Reliability and Security Companion (QRS-C), Prague, 2017, pp. 618-619.

[7] P. Erik Strandberg, W. Afzal, T. J. Ostrand, E. J. Weyuker and D. Sundmark, "Automated System-Level Regression Test Prioritization in a Nutshell," in IEEE Software, vol. 34, no. 4, pp. 30-37, 2017.

[8] S. Hafez, M. ElNainay, M. Abougabal and S. ElShehaby, "Potential-fault cache-based regression test selection," 2016 IEEE/ACS 13th International Conference of Computer Systems and Applications (AICCSA), Agadir, 2016, pp. 1-8.

[9] I. Alagöz, T. Herpel and R. German, "A Selection Method for Black Box Regression Testing with a Statistically Defined Quality Level," 2017 IEEE International Conference on Software Testing, Verification and Validation (ICST), Tokyo, 2017, pp. 114-125.

[10] A. Ammar, S. Baharom, A. A. A. Ghani and J. Din, "Enhanced Weighted Method for Test Case Prioritization in Regression Testing Using Unique Priority Value," 2016 International Conference on Information Science and Security (ICISS), Pattaya, 2016, pp. 1-6.

[11] A. A. Haider, A. Nadeem and S. Akram, "Regression Test Suite Optimization Using Adaptive Neuro Fuzzy Inference System," 2016 International Conference on Frontiers of Information Technology (FIT), Islamabad, 2016, pp. 52-56.

[12] X. Wang and H. Zeng, "History-Based Dynamic Test Case Prioritization for Requirement Properties in Regression Testing," 2016 IEEE/ACM International Workshop on Continuous Software Evolution and Delivery (CSED), Austin, TX, 2016, pp. 41-47.

[13] P. E. Strandberg, D. Sundmark, W. Afzal, T. J. Ostrand and E. J. Weyuker, "Experience

ISSN: 1992-8645 www.jatit.org E-ISSN: 1817-3195

Report: Automated System Level Regression Test Prioritization Using Multiple Factors," 2016 IEEE 27th International Symposium on Software Reliability Engineering (ISSRE), Ottawa, ON, 2016, pp. 12-23.

[14] N. Dini, A. Sullivan, M. Gligoric and G. Rothermel, "The Effect of Test Suite Type on Regression Test Selection," 2016 IEEE 27th International Symposium on Software Reliability Engineering (ISSRE), Ottawa, ON, 2016, pp. 47-58.

[15] Priyanka, H. Kumar and N. Chauhan, "A novel approach for selecting an effective regression testing technique," 2016 3rd International Conference on Computing for Sustainable Global Development (INDIACom), New Delhi, 2016, pp. 1122-1125.

[16] P. Dhareula and A. Ganpati, "Identification of attributes for test case reusability in regression test selection techniques," 2016 3rd International Conference on Computing for Sustainable Global Development (INDIACom), New Delhi, 2016, pp. 1144-1147.

[17] B. Srisura and A. Lawanna, "False test case selection: Improvement of regression testing approach," 2016 13th International Conference on Electrical Engineering/Electronics, Computer, Telecommunications and Information Technology (ECTI-CON), Chiang Mai, 2016, pp. 1-6.

[18] S. Akimoto, S. Nakanishi, R. Yaegashi and T. Takagi, "Extended differential control flow graphs for the selection of test cases in regression testing," 2016 IEEE/ACIS 15th International Conference on Computer and Information Science (ICIS), Okayama, 2016, pp. 1-2.

[19] S. Kumar, P. Ranjan and R. Rajesh, "Modified ACO to maintain diversity in regression test optimization," 2016 3rd International Conference on Recent Advances in Information Technology (RAIT), Dhanbad, 2016, pp. 619-625.

[20] S. Dahiya, R. K. Bhatia and D. Rattan, "Regression test selection using class, sequence and activity diagrams," in IET Software, vol. 10, no. 3, pp. 72-80, 6 2016

[21] P. Kandil, S. Moussa and N. Badr, "A methodology for regression testing reduction and prioritization of agile releases," 2015 5th International Conference on Information & Communication Technology and Accessibility (ICTA), Marrakech, 2015, pp. 1-6.

[22] D. Gao, X. Guo and L. Zhao, "Test case prioritization for regression testing based on ant colony optimization," 2015 6th IEEE International Conference on Software Engineering and Service Science (ICSESS), Beijing, 2015, pp. 275-279.

[23] E. D. Ekelund and E. Engström, "Efficient regression testing based on test history: An industrial evaluation," 2015 IEEE International Conference on Software Maintenance and Evolution (ICSME), Bremen, 2015, pp. 449-457.

[24] N. Chauhan, M. Dutta and M. Singh, "A Program Model Based Regression Test Selection Technique for Object-Oriented Programs," 2015 Fifth International Conference on Communication Systems and Network Technologies, Gwalior, 2015, pp. 918-924.

[25] M. A. El-Din, I. A. E. H. Taha and H. El-Deeb, "Enhanced HGS algorithm (EHGSA) for cost reduction regression testing," 2015 Science and Information Conference (SAI), London, 2015, pp. 921-927.