Cost Optimization in Regression Testing: A Review

Amita Dhankar, Varsha Rani
Asst. Professor, Student
UIET, MDU Rohtak (Haryana), India

Abstract - **Regression testing is very important aspect of software upgradation. Testing every piece of software each time is very expensive due to growth of test suites as software grows. As process of regression testing can be costly and exhaustive, eventually that can impact software delivery time and cost. To solve the cost and delivery time problem in regression testing we need a technique which can optimize the cost and that technique is code graph. In this paper there is a survey presented in context of regression testing techniques and graph related problem in context of clique and independent graph i.e. evaluation of techniques for cost optimization, so that the effective technique can be concluded.**

Index Term - **Regression Testing, Code Graph, Subgraph, Path Complexity, Cost Optimization.**

1. INTRODUCTION

"Software testing is the process of executing the program with the intention of finding error". For this reason we need to generate test cases for implementation to find the error. Maintenance activity takes near about 60% of the cost of overall software production. Studies have shown that cost of correcting a fault at the time of maintenance is 10 times more costly than before it i.e. at the time of software development.

IEEE defines regression testing as follows **[2]**-"Regression testing is discriminatory retesting of a system or component to verify that modifications have not caused unintended effects and that the system or component still complies with its specified requirements."

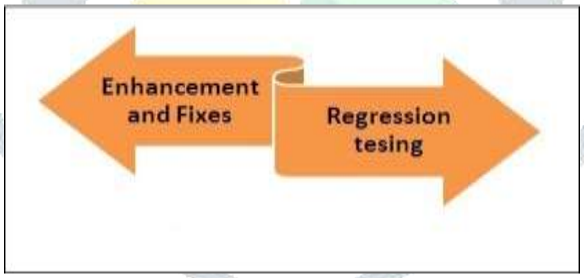


Fig 1: Regression Testing

Regression testing is a activity that is performed to provide confidence that modifications do not harm the existing behavior of the software. Regression testing is performed when changes are performed into an existing system and we need to show that changes don't affect the behavior of the existing system, unchanged part of the system.

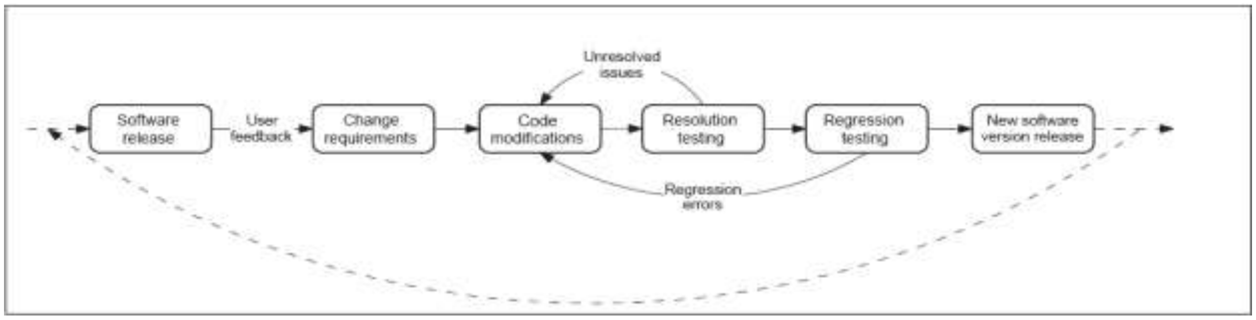


Fig 2: Activities that take place during software maintenance and regression testing

Figure 2 shows the activities takes place in regression testing during the maintenance phase after the release of software. As shown in figure, after software is released, the failure reports and the change requests for the software are compiled, and the software is modified to make necessary changes. Regression testing is usually performed by running some, or all, of the test cases created to test modifications in previous versions of the software. Test suites grow in size as software progress or when we modify the software as per requirement, often making it too costly to execute the entire test cases.

In order to deliver a compatible software product testing is performed. As testing progress increases it directly affect the overall project cost. Many times it happens that the actual cost of software becomes than the estimated cost. Cost is considered the most important parameter with respect to testing in software industry and so as we need to optimize the cost. Code Graph is the technique used to divide the graph. Code Graph is an independent part of the program which does not affect the behavior of remaining program. In Code Graph we divide the graph into sub graph into independent set and finding the minimum clique to make the result more definite and efficient. Using Code Graph we will traverse minimum path to get the optimal path with optimized cost with the help of path complexity. In recent year's researchers have done a variety of work in the area of Cost optimization by using various concepts like Genetic Algorithm, simulated annealing and Automation in generation of test data etc.

2. CODE GRAPH

In, Code Graph technique we divide the graph into subgraphs by finding minimal cliques and independent sets. Independent sets are the part of program which do not affects each other.

Clique - An undirected graph is formed by a finite set of vertices and a set of edges. By convention, the number of vertices in the graph is denoted by *n* and the number of edges is denoted by *m*. A clique in a graph *G* is a complete subgraph of *G*; i.e. it is a subset *S* of the vertices such that every two vertices in *S* are connected by an edge in *G* [22]. A maximal clique is a clique to whom no more vertices can be added; a maximum clique is a clique that includes the largest possible number of vertices.

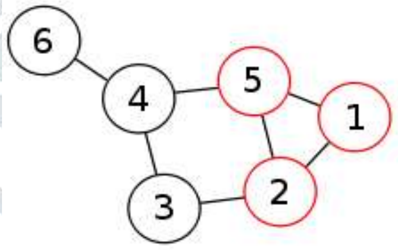


Fig 3: Clique

In figure 3 the graph has one maximum clique, the tringle {1,2,5}, and four maximal clique ,the pairs {2,3}, {3,4}, {4,5} and {4,6}.

Independent Set - In graph theory, an independent set or stable set is a set of vertices in a graph, no two of which are adjacent. That is, it is a set *I* of vertices such that for every two vertices in *I*, there is no edge connecting the two.

In figure 4 the nine blue vertices form a maximum independent set for the generalized graph [23].

Code Graph consists of three steps. First, the preprocessing of the input that normalizes the input into a standard form which is acceptable by the next phase and source code will goes as input in the Code Text Area. Second, an intermediate representation of the input in which the normalized input is transformed into graph by formal approach. Third, the graph is divided into subgraph and calculates the path complexity.

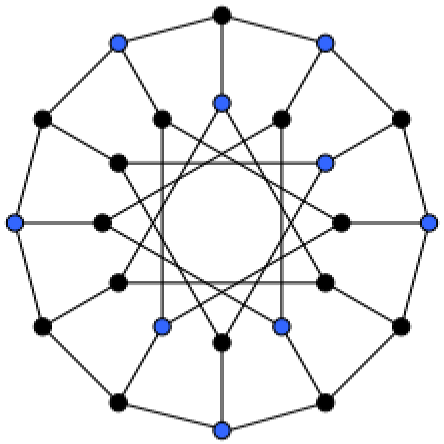
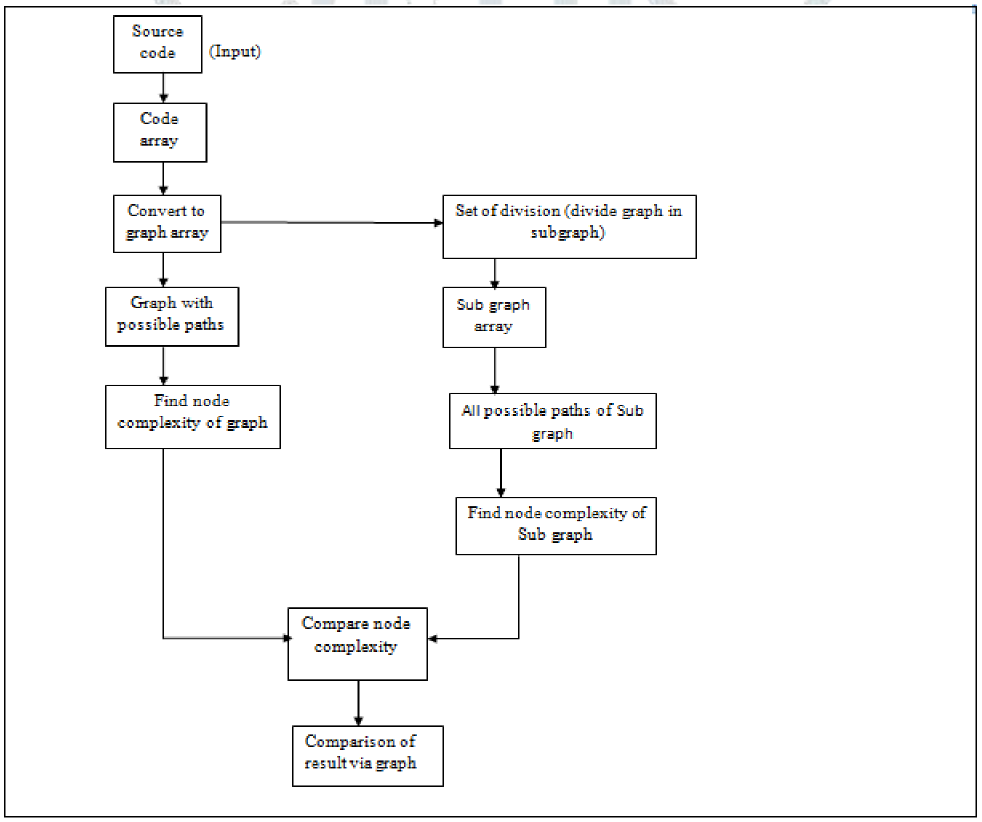


Fig 4: Independent set

Figure 5 shows the complete block diagram of the proposed system. This figure gives a brief overview of activities take place in code graph i.e. complete process of code graph.


Source code
(Input)
Code array
Convert to graph array
Set of division (divide graph in subgraph)
Graph with possible paths
Sub graph array
Find node complexity of graph
All possible paths of Sub graph
Find node complexity of Sub graph
Compare node complexity
Comparison of result via graph
Fig 5: Block diagram of Code Graph

3. LITERATURE SURVEY

Several regression testing techniques are introduced during many years. Various studies have been done in the field of test suite minimization, test case selection and prioritization. The intent of regression testing is to ensure that a change, such as a bug fix, did not introduce new faults. One of the main reasons for regression testing is to determine whether a change in one part of the software affects other parts of the software". To find that we need to perform regression testing and it should be as that optimize the cost and minimize the effort. For the literature survey several papers were studied and tried to get an approach with the help of previous papers, which can optimize the cost.

Hopcroft et al. [1] presented an algorithm for dividing a graph into triconnected components. When it is implemented on a random access computer, the algorithm requires O(V + E) time and space to analyze a graph with V vertices and E edges. The algorithm is both theoretically optimal to within a constant factor and efficient in practice. The algorithm is not only theoretically optimal but practically useful. The split components algorithm has been implemented in ALGOL.

Randy and Panos [3] have shown that a simple algorithm implemented very efficiently can solve very large maximum clique problems. Since this algorithm is very simple to implement, it can be used as a basis for computational comparison when different algorithms are used.

Rothermel and Harrold [4] have discussed issues relevant to selective retest approaches, and present a framework within which such approaches can be evaluated in terms of inclusiveness, precision, efficiency, generality, and accountability. This framework is used to evaluate and compare existing selective retest algorithms. The evaluation reveals strengths and weaknesses of existing methods, and highlights problems that future work in this area can address.

Agrawal gave the term dominators, superblocks and mega blocks for the purpose of implicating coverage among basic blocks for reduction of requirements that are for coverage criteria and apply for a program in context.[5][9].

David R. Wood introduces a branch-and-bound algorithm for the maximum clique problem which applies existing clique finding and vertex coloring to determine lower and upper bounds for the size of a maximum clique. The author have observed that for graphs with fixed size and density the difficulty of the maximum clique problem is generally inversely proportional to the size of a maximum clique in the graph [6].

Rothermel and Harrold told that test selection techniques select subsets that exclude no tests that if executed would reveal faults in the modified software. This paper reports empirical studies on a particular safe regression test selection technique, in which the technique is compared to the alternative regression testing strategy of running all tests cases. The cost of analysis necessary for test selection and the cost of executing and validating the tests interact to affect cost-effectiveness [7].

Wong and his team [8] presented a collective study of them called WHLM study. In their study they examine ten common programs that are written for UNIX. This examination is done by test suites that are generated randomly. For the reduction of test suite they used the tool called ATAC tool. There was a pool of test cases generated by Domain based test suite generator. Authors examine the difficulty in fault detection by categorizing the faults covering by test cases in Quartiles.

Graves et al. presented initial results of an experiential study of regression test selection techniques. This study examined some of the costs and benefits of several regression test selection techniques. Minimization produced the smallest and the least effective test suites. The safe and dataflow techniques had nearly equivalent average behavior in terms of cost-effectiveness, dataflow techniques require at least as much analysis as the two most efficient safe techniques. The safe technique found all faults for which we had fault-revealing test cases while selecting 60% of the test cases on the median [10].

Malishevsky et al. the authors developed cost models for assessing the cost benefits of regression test selection, test suite reduction, and test case prioritization techniques. They primarily considered assessment after the fact of regression testing techniques; this can be used by practitioners to make decisions analyzing historical data, or by researchers to evaluate experimental results [11].

Volker Stix explains that paper provides two algorithms which keep track of all maximal cliques inside dynamic graphs. The experiments show, that this dynamic approach is more efficient than a static one. Clustering applications dealing with perception based or biased data. There, objects to be clustered are allowed to belong to several clusters at the same time which results in a fuzzy clustering. In this article algorithms are provided to track all maximal cliques in a fully dynamic graph [12].

S Elbaum et al. authors have presented APFDc (Average Percentage of Fault Detection with Cost) for assessing the rate of fault detection of prioritized test cases that incorporates varying test case and fault costs. They have also presented techniques for prioritizing test cases that attempt to account for the effects of varying test case cost and fault severity. Their case study applying these techniques illustrates the application of the metric and several effects that arise in the use of different test cost and fault severity distributions. One focus in their work has been the APFDc metric i.e. APFD cost metric [13].

S. Elbaum et al. have shown an alternative approach using data obtained from the application of several prioritization techniques to several substantial programs, they compare the performance of these prioritization techniques in terms of effectiveness and show how the results of this comparison can be used together with cost-benefit threshold information to select a technique that is most likely to be cost-effective. They have proposed two strategies: basic instance and threshold strategy and enhanced instance and threshold strategy [14].

G. Rothermel et al. have presented a study assessing the effects of time-constrained on the costs and benefits of test case prioritization techniques. Their results show that time constraints can indeed play a significant role in determining both the cost-effectiveness of prioritization techniques and the relative cost-benefit trade-offs among techniques. They have performed the Kruskal – Wallis test and Bonferroni test for comparing across techniques per time constraint level and program [15].

S. Yoo and M. Harman described that test suites tend to grow in size as software evolve, often making it too costly to execute entire test suites. A number of different approaches have been studied to maximize the value of the accrued test suite: minimization, selection and prioritization. This paper provides both a survey and a detailed analysis of trends in regression test case selection, minimization and prioritization. This paper shows how to work on these areas and provides a survey on the development of these ideas, their applications, empirical evaluation and open problems for future work [16].

Praveen Ranjan Srivastava1 and Tai- hoon Kim presented a method for optimizing software testing efficiency by identifying the most critical path clusters in a program. They do this by developing variable length Genetic Algorithms that optimize and select the software path clusters which are weighted in accordance with the criticality of the path. Exhaustive software testing is rarely possible because it will become difficult for medium size software. Only parts of a program can be tested, but these parts are not necessarily the most error prone. By identifying the most critical paths, the testing efficiency can be increased. Genetic algorithms are often used for optimization problems. The Genetic Algorithms also outperforms the exhaustive search and local search techniques. In conclusion, by examining the most critical paths first, they obtain a more effective way to approach testing which in turn helps to refine effort and cost estimation in the testing phase [17].

Anu Sharma et al. presented a meta-heuristic search technique that depends upon the neighborhood solution this is unique kind of work what we have proposed in this paper solves the problem of cost optimization in software testing. This paper presents use of tabu search with dijktsra's algorithm (a greedy approach) to provide an efficient path with maximum code coverage and minimum cost. The structure of the paper shows that tabu search depends upon searching the neighboring nodes and memorizing the best solution in its short term memory. All other related solutions are memorized in long term memory of the system which makes tabu search simple to apply in the problem [18].

S. Kadry developed a new technique to improve the cost-effectiveness of the regression testing. An evaluation between the proposed technique and two well known techniques: Automated test using viability method and Test selection based on Risk Analysis is given. The evaluation is based on three factors: execution time, number of detected errors and the deploying time [19].

Bharati et al. provides the analysis of both code-based and model-based regression testing technique according to some comparison and evaluation criterion. They focused on optimal selection of subset of test cases from the initial test suite to minimize the testing time, cost and effort. This paper contains a survey that presents code-based and model-based regression testing and their analysis with respect to the parameters presented by Harrold. Model based regression testing techniques can be helpful to improve the exiting techniques [20].

Nitika Sharma et al. have described the hybrid criteria's in different prospective with existing techniques. Selecting and choosing minimum number of test cases according to result is their major goal. In their work they have formulized the swarm algorithm for hybrid criteria. Hybrid criteria use Rank, Merge and Choice for building the test cases from test suite for minimizing the redundancy. This research will lead to give better efficiency in regression testing using hybrid criteria. Path Coverage deals with the test case selection as it gives all the details of test cases. Result of this paper shows the effective priority of finding the faults by covering all the criteria's together [21].

4. COST OPTIMIZATION TECHNIQUES

Regression Testing is an emergent area. It is performed whenever any alteration has been put up into a system. Whenever any testing activity is performed there is a process of executing test cases. In order to deliver a compatible software product testing is performed. As testing progress increases it directly affect the overall project cost. The size of software increases with respect of time when changes are performed, so size of test suite also changes [4] [7]. Regression testing becomes very difficult with such large suite in terms of reliability and efficiency. Many times it happens that the actual cost of software becomes than the estimated cost [10]. Cost is considered the most important parameter with respect to testing in software industry and so as we need to optimize the cost. For the sake of time and cost there are various techniques are discovered to reduce size and to minimize the cost.

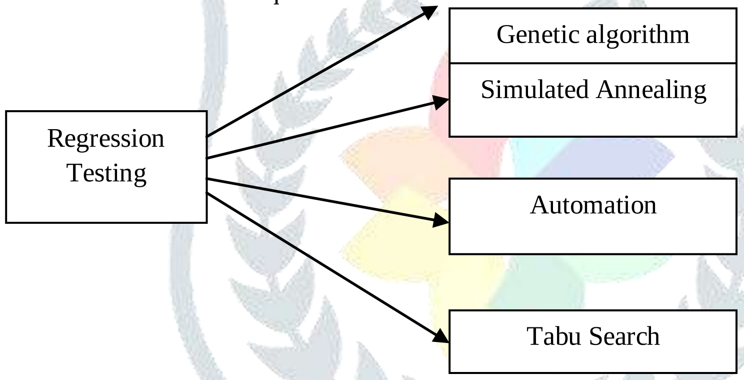


Fig 6: Cost Optimization Techniques used in Regression Testing

In recent year's researchers have done a variety of work in the area of Cost optimization by using various concepts like Genetic Algorithm, simulated annealing and Automation in generation of test data etc.

Code Graph is the technique used to divide the graph. A Code Graph is a graph in which the vertices can be partitioned into a clique and independent set. Code Graph is an independent part of the program which does not affect the behavior of remaining program. In Code Graph we divide the graph into sub graph into independent set to make the result more specific and effective. Using Code Graph we will traverse minimum path to get the optimal path with optimized cost.

5. PATH COMPLEXITY

Path complexity is calculated for each possible path of the graph and path complexity of whole graph and subgraph. Initial path complexity is assumed to 0 (zero) in the starting. Path complexity is sum of path length of all possible paths in graph from start to end node.

Initial path complexity =0;

PC for whole graph/subgraph = $\sum_{j,i=1}^{N} \boldsymbol{no. of\ nodes\ in\ P_j\ \ /S_i}$

N is the total no of paths in a graph/subgraph.

Complexity of each node that is included in path is assumed to be 1 which will be added during the calculation of path complexity.

No. of nodes in any Path = Node 1 + Node 2 + Node 3 +…………+ Node N.

N is the total number of nodes in that path.

Let us assume we are having a path i.e. Path 1= 1-2-3-4-6-7-8-10-11-15

Path1 = 1+ 2+3+4+6+7+8+10+11+5;

Where 1, 2, 3…etc are node name and complexity of each node is assumed to be 1 so no. of nodes in path will be

No. of nodes in Path 1=1+1+1+1+1+1+1+1+1 = 9

6. CONCLUSION

Regression testing is a kind of testing that help developer to make sure that no defects has occurred after the performing the modification. This survey examined some of the costs and benefits of several regression test selection techniques. Results, although preliminary, highlight several differences among the techniques, expose essential trade-offs, and provide an infrastructure for further research by ourselves and others. There are several techniques used to in regression testing. Since testing is complex and there is no direct measure of fault exposure likelihood and there are many different types of cost involved in it. A certain regression testing technique cannot be used at each and every scenario and there exists a need to find a technique that can be used in a particular framework every time the scenario is obtained. This paper contains the literature survey of the cost optimization technique that was used in the industry.

We can work on sophisticated and improved algorithms to get more efficient results. The new algorithm is based on these research techniques but with new innovative idea so that we can reduce the effort, cost and time of development process.

REFERENCES

[1] J. E. Hopcroft And R. E. Tarjan, 1973, Dividing A Graph Into Triconnected Components, *Computer Science Department, Cornell University, Ithaca, New York 14850*, pp 135-158.

[2] IEEE Standard Glossary of Software Engineering Terminology, 1983, *IEEE Std* , pp-729-1983, IEEE Press.

[3] Randy Carraghan and Panos M. Pardalos, 1990, an Exact Algorithm for the Maximum Clique Problem, *Elsevier Science Publishers B.V. Operations Research Letters*, pp 375-382.

[4] Gregg Rothermel and Mary Jean Harrold, 1994, A Framework for Evaluating Regression Test Select ion Techniques, *IEEE*, pp 201-210.

[5] H. Agrawal, 1994, Dominators, super blocks, and program coverage, *21st ACM SIGPLAN SIGACT symposium on Principles of Programming Languages, Portland, Oregon.*

[6] David R. Wood, 1997, an algorithm for finding a maximum clique in a graph, *Operations Research Letters 21, Elsevier Science B.V*, pp 211-217.

[7] Gregg Rothermel and Mary Jean Harrold, 1998, Empirical Studies of a Safe Regression Test Selection Technique, IEEE *TRANSACTIONS ON SOFTWARE ENGINEERING, VOL. 24, No. 6,* pp 401-419.

[8] W. E. Wong, J. R. Horgan, 1998, S. London, and A. P. Mathur, Effect of test set minimization on fault detection effectiveness, *Software Practice and Experience*, 28(4):347,369.

[9] H. Agrawal, 1999, Efficient Coverage Testing Using Global Dominator Graphs, *ACM SIGPLAN-SIGSOFT Workshop on Program Analysis for Software Tools and Engineering, Toulouse, France*.

[10] Todd L. Graves and Gregg Rothermel, 2001, an Empirical Study of Regression Test Selection Techniques, *ACM Transactions on Software Engineering and Methodology*, pp 184–208.

[11] Malishevsky, A., Rothermel, G. and Elbaum, 2002, Modelling the Cost Benefits Trade-offs for Regression Testing Techniques, *IEEE International Conference on Software Maintenance.*

[12] Volker Stix, 2004, Finding all maximal cliques in Dynamic Graphs, *Computational Optimization and Applications*, 27, pp 173-186.

[13] Elbaum, S, Malishevsky, A. and Rothermel, 2001, Incorporating Varying Test Costs and Fault Severities into Test Case Prioritization. *Proceedings of the International Conferenece on Software Engineering.*

[14] Elbaum, S, et al. s.l. , 2004, Selecting a cost effective test case Prioritization Techniques.: *Software Quality Journal.*

[15] Gregg Rothermel and Mary Jean Harrold, 2007, A Safe, Efficient Regression Test Selection Technique, *ACM Transactions on Software Engineering and Methodology, Vol. 6, No. 2,* Pages 173–210.

[16] S. Yoo, M. Harman, 2007, Regression Testing Minimization, Selection and Prioritization: A Survey, *Software Testing, Verification and Reliability,* Pages 1- 56.

[17] Praveen Ranjan Srivastava1 and Tai-hoon Kim, 2009, Application of Genetic Algorithm in Software Testing, *International Journal of Software Engineering and Its Applications Vol. 3, No.4*, pp - 87-96.

[18] Anu Sharma, ArpitaJadhav, Praveen Ranjan Srivastava, Renu Goyal, 2010, Test Cost Optimization Using Tabu Search, *J. Software Engineering & Applications,* pp. 477-486 .

[19] Kadry, 2011, A New Proposed Technique to Improve Software Regression Testing cost, *international Journal of Security and its Applications*.

[20] Chandana Bharati, Shradha Verma, 2013, Analysis of Different Regression Testing Approaches, *International Journal of Advanced Research in Computer and Communication Engineering Vol. 2, Issue 5,* Pages 2150-2155.

[21] Nitika Sharma and Neha Malhotra, 2014, Regression Testing Prioritization, Selection and Reduction using Hybrid Criteria, *International Journal of Computer Applications (0975 – 8887),* pages 38-46.

[22] https://en.wikipedia.org/wiki/Clique_problem

[23] https://en.wikipedia.org/wiki/Independent_set