On the Applicability of Soft Computing Techniques in Regression Testing

Esha Khanna
Assistant Professor
Department of Information Technology
D. A. V Institute of Management, India

Ketna Khanna
M. Tech Scholar
Department of Computer Engineering
YMCA University of Science and Technology, India

ABSTRACT

Regression testing is required whenever changes are incorporated in software. Regression testing re-executes the test cases to verify that changes made during maintainance phase have not caused any unintended effects. Test case prioritization technique re-orders the test cases in such a way that important test cases are executed within the resources. The work reviews different test case prioritization techniques. The work also presents computational intelligence based test case prioritization techniques. The work reviews test case prioritization based on Neural Networks, Genetic Algorithms and Fuzzy Logic.

General Terms

Testing, Verification, Algorithm.

Keywords

Regression Testing, Computational Intelligence, Test Case Prioritization.

1. INTRODUCTION

During maintenance phase, software undergoes various changes. These changes include changed user requirements, enhanced functionality and fixing of defects. Regression testing is a process of re-executing test cases to verify that changes introduced in software has not caused unexpected consequences. Regression testing is the selective re-testing of a system or component to verify that modifications have not caused unintended effects and that the system or the component still complies with its specified requirements [1].

Regression testing is a costly process. Due to constrained resources, execution of all the test cases is not possible. Techniques of regression testing are minimization, selection and prioritization [2]. Test case minimization and selection reduces the number of test cases of regression test suite while test case prioritization reorders the test cases according to some criteria such that most important test cases are executed prior to others.

The work reviews various test case prioritization techniques. An extensive literature review has been carried out using the guidelines given by Kitchenham [3]. The work presents various computational intelligence based test case prioritization techniques. Computational Intelligent techniques are inspired by natural human processes. These include Neural Networks, Genetic Algorithms and Fuzzy Logic. The paper surveys work undertaken in test case prioritization using computational intelligent techniques.

The goals of this paper are

- To present various test case prioritization techniques.
- To review computational intelligence based test case prioritization techniques.

The paper has been organized as follows. Section two presents regression testing. Section three discusses test case prioritization in detail. Section four reviews computational intelligence based test case prioritization techniques and section five concludes.

2. REGRESSION TESTING

As the software evolves, new functionality may be added or some existing functionality may be modified. Changes made in one module of the software may affect other unchanged modules. Therefore, there arises a need to re test the complete software every time when changes are incorporated. Regression testing calls for re execution of test cases in order to verify that changes made in the software has not caused any unexpected consequences. Regression testing is selective retesting of the system or components to verify that modifications have not caused an unintended effects and the system or component still complies with its specified requirements [4].

Test cases in regression testing may be classified as new, re-useable and obsolete. Regression testing can be carried out manually or can be automated by using tools like selenium and Quick Test Professional [5].

Regression testing is required in following scenarios [6]

- Changed requirements
- Enhanced functionality
- Defect fixing

Techniques of regression testing are re-test all, test case minimization, test case selection and test case prioritization [2]. Re-test all technique re-executes all the test cases in order to verify that changes made in software have not caused unexpected consequences. It is a costly technique. All the test cases cannot be re-executed due to limited resources.

Test case selection technique selects test cases from large pool which are associated with changed modules [2]. Test case minimization technique minimizes the test cases based on same criteria such as coverage. Test case minimization techniques aim to identify redundant test cases and to remove them from the test suite in order to reduce the size of the test suite [2].

Test case prioritization technique reorders the test case the test cases of regression test suite based on some given criteria in such a way that important test cases are executed within the resources [2, 7].

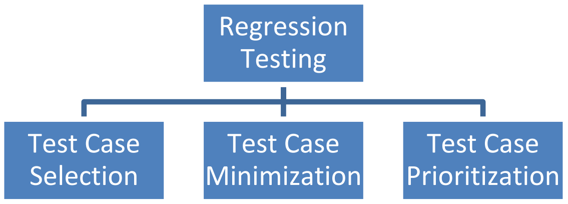

Fig 1: Regression Testing Techniques

Regression testing is a costly process. Some of the challenges of regression testing are as follows.

- As the software evolves, test cases become colossal.
- Minimization of test suite while achieving maximum coverage.
- Frequency of regression testing.

3. TEST CASE PRIORITIZATION

In regression testing, number of test cases may become colossal. Due to limited amount of resources, all these test cases cannot be executed. Further, all test cases are not equally important. Test cases are therefore prioritized in such a way that test cases that are able to detect more number of faults are executed first. Test case prioritization techniques re-order the test cases according to some criteria in such a way that important test cases are executed prior to others [8]. Prioritization criteria may be based on test case history [9], coverage [10], fault severity [11], customer requirements [12] or costs [13].

Test case prioritization techniques can be classified as follows. [14]

- No prioritization
- Random prioritization
- Optimal prioritization
- Total statement coverage
- Additional statement coverage
- Total branch coverage
- Additional branch coverage
- Total fault exposing potential prioritization
- Additional fault exposing potential prioritization

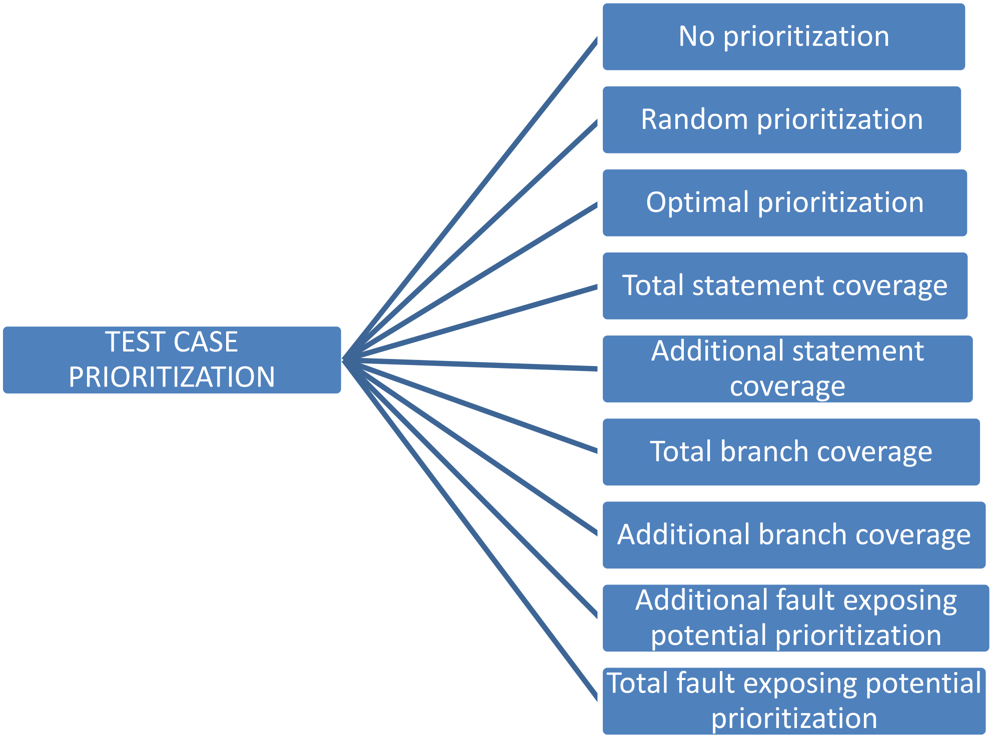
TEST CASE PRIORITIZATION
No prioritization
Random prioritization
Optimal prioritization
Total statement coverage
Additional statement coverage
Total branch coverage
Additional branch coverage
Additional fault exposing potential prioritization
Total fault exposing potential prioritization

Fig 2: Test Case Prioritization Techniques

4. CI BASED REGRESSION TECHNIQUES

4.1 Artificial Neural Networks

Artificial neural networks are soft computing techniques which mimics the human brain in order to perform the given task. Artificial Neural networks (NN) are physical cellular systems which can acquire, store and process the experiential knowledge [15]. Neural networks perform the task by using the experiential knowledge i.e examples or history. Neural networks are used for neural processing like autoassociation, hetroassociation, classification and recognition [15]. Artificial neural networks are used for recognition and regression. Artificial neural networks have been used in regression testing for prioritizing the test cases. Some of the works has been presented in this section.

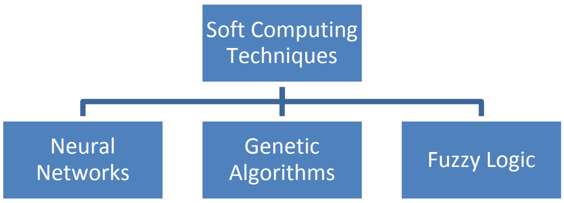

Fig 3: Soft Computing Techniques

The work by Wang, et. al. [16], proposed a framework for test case prioritization using machine learning and program slicing. The work combined machine learning and program slicing and proposed a program for test case prioritization. Test cases were generated using a Library Intelligent test oracle library. Artificial Neural Networks were used to learn the count that a program segment will be visited. The work was experimented on two benchmarks program of software artifact infrastructure repository. The results showed increased rate of fault detection.

The work proposed by Gokce, N., Eminli, M. uses multilayer perceptron neural network in order to prioritize the test cases using model based prioritization [17]. The work was implemented on 100 test cases, which were derived from ESG of web based software system. MATLAB tool was used to construct a multilayer perceptron network with input layer having, 190(95*2) neurons. Output layer of neural networks consisted of 5 neurons representing priority groups. Training was performed using Levenberg-Marquardt training function. 70% data set was used for training, 15% for validation and rest 15% for testing. The results showed the accuracy of about 96%.

The work by Bhasin, et. al. proposed a neural network based model that assigns priority to test cases [18]. The work also discussed the policies to prioritize test cases. The work was implemented on 200 test cases using backpropogation network. The experiment was conducted using 2 layer, 5 layer, 10 layer, 15 layer, and 20 layer neural network. The results showed that neural network can predict accurate priority of test cases.

Gokce et. al. presented coverage based test case prioritization using unsupervised neural network [19]. The neural network was trained by adaptive competitive learning algorithm. The work has been validated on a case study.

In the work by Simao, et. al. [20], ART-2A self organizing neural networks were used in order to carry out test case reduction in regression testing. Neural networks classify the test cases into relevant classes based on feature vector. The work compared proposed technique which used feature vectors to a random selection approach. Results showed the efficiency of new proposed technique over random selection.

Table 1. Neural Network based techniques

Author	Technique	Results
Wang, et. al. [16]	Machine learning and program slicing	Proposed technique was implemented on two benchmark programs of software artifact infrastructure repository. The technique resulted in increased rate of fault detection.
Gokce, N., Eminli, M. [17]	Multilayer perceptron neural network. The work was implemented using MATLAB.	The technique was implemented on 100 test cases crafted for ESG of web based software system. The technique showed the accuracy of 96%.
Bhasin, H. et. al. [18]	Backpropogation neural network. Levenberg-Marquardt backpropogation ('TRAINLM' training function) is chosen	The technique was implemented on 200 test cases using 2 layer, 5 layer, 10 layer, 15 layer and 20 layer neural network. Results showed the average error of 0.16%.
Gokce et. al. [19]	Unsupervised Neural Network.	Adaptive competitive learning algorithm was used for training. The work was validated on a case study.
Simao, et. al. [20]	ART-2A self organizing Neural Networks.	The work compared proposed technique to random selection approach. Results showed increased efficiency.

4.2 Genetic Algorithms

Genetic algorithm (GA) is heuristic search algorithm inspired by Charles Darwin theory of natural evolution. GA is an intelligent search technique and is used to solve optimization problems [21]. An extensive literature review has been carried out in order to find out the techniques which use Genetic algorithms to carry out the task of regression testing.. Some of the works has been summarized in this section.

Li, Z. et. al., compared the performance of genetic algorithms and greedy approaches in order to prioritize regression test cases [22]. The techniques were applied to six programs ranging 374 to 11,148 lines of codes. The work showed the effectiveness of metaheuristic and evolutionary search algorithms to prioritize test cases. The work used three choices of fitness metric and addressed problems of fitness matrix, suitable search technique and landscape modality.

Konsaard, P., Ramingwong, L. presented a genetic algorithm based test case prioritization technique that prioritizes test cases on the criteria of total coverage [23]. Test cases that covered maximum code were assigned higher priority than others. The work also compared genetic algorithm with five other approaches on the performance of average percentage of condition covered and execution time.

In another work, genetic algorithm was used to prioritize test cases on statement coverage criteria [24]. Average percentage of statement coverage (APSC) metric was used to show the efficiency of proposed algorithm.

In the work by Khanna, E., genetic algorithm reorders regression test cases on criteria of maximum fault detection [25]. Fitness value of test cases is evaluated using number of fault covered. Based on fitness value, an optimal sequence of test cases is selected which leads to total fault coverage.

Bhasin, H., Manoj, proposed a Genetic Algorithm based prioritization technique [26]. Coupling number calculator was used in order to assign fitness value to test cases. Test cases prioritized using this technique resulted in high fault detection rate in comparison to others.

Huang, Y. C., et. al. proposed a cost cognizant prioritization technique which takes into account test case costs and fault severities in test case prioritization [27]. The work used genetic algorithm and test historical records to prioritize the test cases of regression test suite. The results reported the effectiveness of proposed technique by using APFDC (average percentage of faults detected per cost).

The work by Suman, Seema proposed a genetic algorithm based prioritization technique which uses complete code coverage criteria in order to prioritize the test cases dynamically. The work used PMX and cyclic crossover to generate new test cases [28]. The work aimed at reducing the number of test cases of regression suite case without affecting the software quality.

Kaur, A. and Goyal, S., proposed a genetic algorithm based prioritization technique that automates the test case prioritization process based on code coverage [29]. The work presented the effectiveness of proposed algorithm by APCC (average percentage of code covered) metric.

The work by Mishra, P. K., Pattanaik, B.K.S.S , prioritized the regression test cases based on fault severity and customer requirement [30]. The work proposed a genetic algorithm based technique which uses testing time and code coverage information to prioritize test cases. The work also compared genetic algorithm approach with goal oriented and path oriented prioritization technique. The results depicted that genetic algorithm based approach have greater code coverage in less time.

Table 2. Genetic Algorithms based techniques

Author	**Technique**	**Results**
Li, Z., et. al. [22]	Genetic Algorithm and greedy approaches. The work addressed problems of fitness matrix, suitable search techniques and landscape modality.	The technique was applied to 6 programs ranging 374 to 11148 lines of code. The result showed the effectiveness of metaheuristic and evolutionary search algorithm for test case prioritization.
Konsaard, P., Ramingwong, L. [23]	Total code coverage based test case prioritization using genetic algorithms.	The work compared the genetic algorithm technique with five other approaches on the basis of average percentage of condition covered and execution time.
Yadav, D. K., Dutta, S. [24]	Statement coverage based test case prioritization using genetic algorithms.	The efficiency of proposed technique was showed by using APSC (Average percentage of statement coverage) metric.
Khanna, E. [25]	Maximum fault detection based test case prioritization using genetic algorithms.	The technique evaluated the fitness value of test cases based on number of faults covered. Results showed increased rate of fault coverage.
Bhasin, H., Manoj [26]	Coupling number calculator was used to assign fitness value.	The work resulted higher rate of fault detection.
Haung, Y. C., et. al. [27]	Cost cognizant prioritization technique that takes into account test case costs and fault severities.	The results reported the effectiveness using APFDC metric.
Suman, Seema [28]	Code coverage based test case prioritization based on genetic algorithm. PMX and cyclic crossover were used to generate new test cases.	The work aimed at reducing number of test cases without affecting the software quality.
Kaur, A., Goyal, S. [29]	Automated genetic algorithm based test case prioritization technique based on code coverage.	Effectiveness of proposed algorithm was presented by APCC (Average percentage of code covered) metric.
Mishra, P. K., Pattanaik, B.K.S.S. [30]	Genetic algorithm technique based on fault severity and customer requirement.	The work compared genetic algorithm approach to goal oriented and path oriented prioritization. The results showed increased efficiency.

4.3 Fuzzy Logic

Yadav and Dutta [31] presented a fuzzy logic based test case prioritization algorithm. Test case prioritization was done on the criteria of fault detection rate, execution time and requirement coverage. The work was designed for object oriented program.

Haider et. al. [32] used Computational Intelligence based approaches for optimizing the test suite for regression testing. The optimized test suite was further analyzed for safe reduction using control flow graphs. The Computational Intelligence approaches used were evolutionary computation, fuzzy logic, neural network and swarm optimization. The results showed that only fuzzy logic technique returned safe reduction test suite and other techniques are inadequate for regression testing.

In another work, Haider et. al., proposed an expert system that uses fuzzy logic in order to optimize test suite for regression testing [33]. Trade off was achieved among technique used, quality aspects (performance, throughput and code coverage) and testing.

The work by Anwar and Ahsan proposed regression test suite optimization using fuzzy logic (Sugeno model) [34]. The proposed work was implemented on two case studies. The results showed that execution time and size of regression test suite may be reduced upto 50% by using fuzzy logic.

Table 3. Fuzzy Logic based techniques

Author	Technique	Results
Yadav, D.K., Dutta, S. [31]	Fuzzy logic based test case prioritization technique based on fault detection rate, execution time and requirement coverage.	The work was designed for object oriented program.
Haider, et. al. [32]	Computational Intelligence based approaches (Evolutionary computation, Fuzzy Logic, Neural Network and Swarm Optimization)	The results showed that fuzzy logic technique resulted in safe test suite reduction.
Haider, et. al. [33]	Fuzzy logic based Expert System	The proposed technique aimed to achieve tradeoff among performance, throughput and code coverage.
Anwar, Z. , Ahsan, A. [34]	Fuzzy logic (Sugeno Model) based test suite optimization technique.	The work was implemented on two case studies. Results showed that execution time and size of test suite may be reduced upto 50 % by fuzzy logic.

5. CONCLUSION

Modification of software requires re-execution of all the test cases in order to validate that changes in one module have not affected correct functionality of others. Due to constrained resources, all the test cases cannot be re-executed within the development period.   Prioritization of test cases is required in order to maintain quality of regression testing. Test cases need to be prioritized in such a way that important test cases are executed before completion of software. This results in cost effective regression testing within limited time and resources. The work reviews various test case prioritization techniques. The paper presents use of soft computing techniques (Artificial Neural Networks, Genetic Algorithms and Fuzzy Logic) in the task of regression test case prioritization. The techniques discussed would be helpful to both researchers and practitioners if the testing has to be stopped prematurely due to lack of resources.

6. REFERENCES

[1] Chauhan, N. 2010. Software Testing principles and practices. Oxford University Press.

[2] Yoo, S., Harman, M. 2012. Regression testing minimization, selection and prioritization: a survey. Software Testing, Verification & Reliability. John Wiley and Sons Ltd. Volume 22 Issue 2, Pp. 67-120.

[3] Kitchenham, B.A. et. al. 2010. Systematic literature reviews in software engineering .A tertiary study, Information & Software Technology .INFSOF , vol. 52, no. 8, pp. 792-805, 2010

[4] IEEE std. definition of Regression Testing.

[5] Pressman, R.S. 2010. Software engineering: a practitioner's approach. McGraw-Hill Higher Education

[6] Rajal, J. S., Sharma, S. 2015. A Review on Various Techniques for Regression Testing and Test Case Prioritization. International Journal of Computer Applications. Volume 116, No. 16.

[7] Srivastava, P., R. 2008. Test Case Prioritization. Journal of Theoretical and Applied Information Technology.pp 178-181

[8] Rothermel, G., R. Untch, C. C., & Harrold, M. 1999. Test Case Prioritization: An Empirical Study. Proc. Int',l Conf. Software Maintenance. IEEE, 179-188.

[9] Engström, E., et. al. 2011. Improving Regression Testing Transparency and Efficiency with History-Based Prioritization -- An Industrial Case Study. Software Testing, Verification and Validation (ICST). IEEE Fourth International Conference on 21-25 March 2011 Page(s):367 - 376 Berlin,IEEE, DOI:10.1109/ICST.2011.27.

[10] David Leon and Andy Podgurski. 2003. A Comparison of Coverage-Based and Distribution-Based Techniques for Filtering and Prioritizing Test Cases. In Proceedings of the 14th International Symposium on Software Reliability Engineering (ISSRE '03). IEEE Computer Society, Washington, DC, USA, 442

[11] Sejun Kim and Jongmoon Baik. 2010. An Effective Fault Aware Test Case Prioritization by Incorporating a Fault Localization Technique. In Proceedings of the 2010 ACM-IEEE International Symposium on Empirical Software Engineering and Measurement (ESEM '10). ACM, New York, NY, USA, Article 5, 10 pages

[12] R. Kavitha, V.R. Kavitha, N. Suresh Kumar. 2010. Requirement Based Test Case Prioritization. In Proceedings of International Conference on Communication Control and Computing Technologies, 826–829.

[13] Ramasamy, K., Mary, S.A., 2008.Incorporating varying requirement priorities and costs in test case prioritization for new and regression testing IEEE, Computing, Communication and Networking.

[14] Elbaum, S., Malishevsky, A. G., & Rothermel, G. 2000. Prioritizing test cases for regression testing. Proceedings of the 2000 ACM SIGSOFT international symposium on Software testing and analysis. ACM, 102-112

[15] Zurada, J.M., Introduction to Artificial Neural Systems, Jaico Publishing House.

[16] Wang, F. et. al. Regression Testing Based on Neural Networks and Program Slicing Techniques the Advances in Intelligent and Soft Computing book series (AINSC, volume 124), IEEE.

[17] Gokce, N. , Eminli, M. 2014. Model based test case prioritization using neural network classification. Computer Science and Engineering: An international journal. Vol 4, No. 1. Pp 15-25.

[18] Bhasin, H. et. al. 2015.Neural Networks based Automated Priority Assigner. Series Advances in Intelligent Systems and Computing. Springer. Volume 381.

[19] Gokce, N. et. al. 2006. Coverage- Based, Prioritized Testing Using Neural Network Clustering. Computer and Information Sciences- ISCIS 2006. Springer.

[20] Simao, A.D.S. et. al. 2006. A Technique to Reduce the Test Case Suites for Regression Testing Based on a Self-Organizing Neural Network Architecture. Computer Software and Applications Conference. COMPSAC '06.IEEE.

[21] Klir, G. J., Yaun, B. 1995. Fuzzy sets and Fuzzy logic theory and applications. Appendix B- Genetic Algorithms- An overview. Pearson Education Inc.

[22] Li. Z., et. al. 2007. Search Algorithms for Regression Test Case Prioritization. Software Engineering, IEEE transactions vol 33 issue 4.

[23] Konsaard, P., Ramingwong, L. 2015. Total coverage based regression test case prioritization using genetic algorithm. 12th International conference on Electrical Engineering/Electronics, Computer, Telecommunications and Information Technology. IEEE.

[24] Yadav, D. K., Dutta, S. 2016. Regression test case prioritization technique using Genetic Algorithm. Advances in Intelligent Systems and Computing book series AISC, volume 509.

[25] Khanna, E. 2016. Regression Testing based on Genetic Algorithms. International Journal of Computer Applications

[26] Bhasin, H., Manoj. 2012. Regression Testing Using Coupling and Genetic Algorithms. International Journal of Computer Science and Information Technologies. Vol. 3(1).

[27] Huang, Y. C., et. al. 2010. Design and Analysis of Cost-Cognizant Test Case Prioritization Using Genetic Algorithm with Test History. Computer Software and Applications Conference (COMPSAC). IEEE.

[28] Suman and Seema. 2012. A Genetic Algorithm for Regression Test Sequence Optimization. International Journal of Advanced Research in Computer and Communication Engineering, Vol. 1, Issue 7

[29] Kaur, A., Goyal, S. 2011. A Genetic Algorithm for Fault based Regression Test Case Prioritization. International Journal of Computer Applications, volume 32, number 8.

[30] Mishra, P. K.,Pattanaik, B.K.S.S. 2010. Analysis of test case prioritization in regression testing using genetic algorithm. International Journal of Computer Applications, volume 75, number 8.

[31] Yadav, D.K., Dutta, S. 2016. Test case prioritization technique based on early fault detection using fuzzy logic. Computing for Sustainable Global Development (INDIACom). IEEE.

[32] Haider, A. et. al. 2016. Safe regression test suite optimization: A review. Open Source Systems & Technologies (ICOSST). IEEE.

[33] Haider, A. et. al. 2012. Test suite optimization using fuzzy logic. International conference on Emerging Technologies (ICET). IEEE.

[34] Anwar,Z.Ahsan,A.2013.Multi-objective regression test suite optimization with Fuzzy logic. Multi Topic Conference(INMIC).IEEE.