Ant Colony Optimization based Test Case Prioritization

[1]Esha Khanna, [2]Ketna Khanna
[1]Assistant Professor, [2]M. Tech Scholar
[1] Information Technology,
[1]DAV Institute of Management, Faridabad, India

Abstract : Regression testing calls for re-execution of old test cases to ensure the correct working of existing features of the software. Due to limited time and resources, all the test cases cannot be executed. Prioritization of test cases is necessary so that higher priority test cases are executed first. Thorough testing increases overall quality of software and reduces maintenance cost. Ant colony optimization (ACO) is a metaheuristic technique used to solve optimization problems. ACO mimics the working of ant colonies to find the best solution to given search problem. ACO has been applied in software testing in order to prioritize the test cases. The work reviews various test case prioritization techniques based on Ant Colony Optimization.

***IndexTerms* – Test case prioritization, Ant colony optimization, Regression testing.**

I. INTRODUCTION

Software testing is a technique for evaluating product quality by identifying the defects and problems [1]. Software testing is the process of executing a program or system with the intent of finding errors [2]. Effective testing ensures the quality of the software and reduces the maintanace costs. After development, many changes may be incorporated in software. New functionality are added and some obsolete features are removed. Changes made in one module may affect the correct working of another module. Therefore, old test cases are reexecuted to ensure that changes have not affected existing featues and code. This is reffered as Regression testing. 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 [3]. Techniques of regression testing include selection, minimization and prioritization. Test case selection and minimization reduces the number of test cases in regression suite while test case prioritization reorders the test cases by keeping the same number of test cases in regression suite. Various soft computing techniques have been used in order to prioritize test cases of regression test suite. These techniques includes Neural Networks [4], fuzzy logic [5], genetic algorithms [6,7,8]. Different search based techniques have been applied to find best order for test case prioritization problem. One such technique is ant colony optimization (ACO). The work reviews various test case prioritization techniques using ACO. An extensive literature review has been carried out according to the guidelines of kitchenham [9]. The work paves the way of ant colony optimization in regression testing.
The paper has been organized as follows. Section two discusses test case prioritization. Section three explains ant colony optimization. Section four reviews various ant colony optimization based techniques of test case prioritization. Section five concludes.

II. TEST CASE PRIORITIZATION

Regression testing re-executes the test cases to validate that changed modules have not interrupted the proper working of other modules [10]. Due to limited time and resources, all the test cases may not be executed. In order to assure quality testing, in constrained environment, test cases are prioritized such that test cases having high probability of fault detection are executed before others. Test case prioritization technique reorders the test cases based on some criterion in such a way that most important test cases are executed within time and resources [11].Test cases may be prioritized on the basis of test case history [12], coverage [13], fault severity [14], customer requirements [15] or costs [16]. Test cases can be prioritized as high, medium and low. Highest priority is assigned to the test cases that must be executed to complete the testing procedure. Such test cases have the highest probability to uncover an error. Medium priority is assigned to the test cases that are executed if time and testing resources are available. Low priority is assigned to the test cases which do not affect the quality of software.

III. ANT COLONY OPTIMIZATION

ACO is a metaheuristic technique used to solve optimization problems. It was proposed by Dorigo in 1990's [17] . It is inspired from the behavior real ant colonies. ACO is based the ability of ants search the shortest path to their food. This phenomenon of ants to find shortest path from their nest to food is known as stigmergy [17].The ACO technique has been used to solve various combinatorial optimization problems. They have been applied to Travelling salesman problem, knapsack problem, data mining, telecommunication network, vehicle routing, test data generation [18,19,20]. The work by K. Doerner[21] and F. P. MicMinn [22] in year 2003 paved the way of ACO to software testing. In one of the works by Bharti suri , an extensive literature review was carried out to find the application of ACO in software testing. (23).
It is a well known fact that ants are blind. They communicate within the colony using the concept of Stigmergy [17, 23]. ACO follows two processes –Pheromone deposition and trail pheromone evaporation.

Ants deposit a chemical known as pheromone as they travel from their nest to food source. This pheromone is detected by other ants of the colony and they tend to follow the path with higher concentrations of pheromone. These ants keep on adding the pheromone on the paths they follow.

1. Ants move in random order from their nest towards food source.
2. As they move towards their goal, they deposit a substance known as pheromone on their path.
3. The choice of paths of other ants is influenced by pheromone as they are able to smell the pheromone.
4. More ants tend to follow the path having higher concentration of pheromone trail.
5. The pheromone trail gradually evaporated with time at some specific rate of evaluation.
6. Ants that follow the shortest path have higher probability of returning back to their nest first.
7. After some time period pheromone levels at shortest path increases and all ants of the colony tends to follow the shortest path.

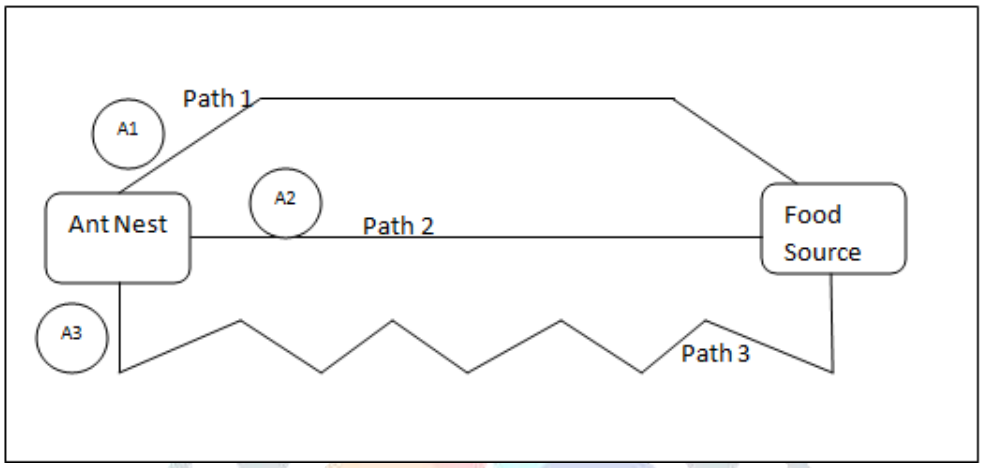

Figure 3.1. Working of ACO, initially 3 ants start from their nest

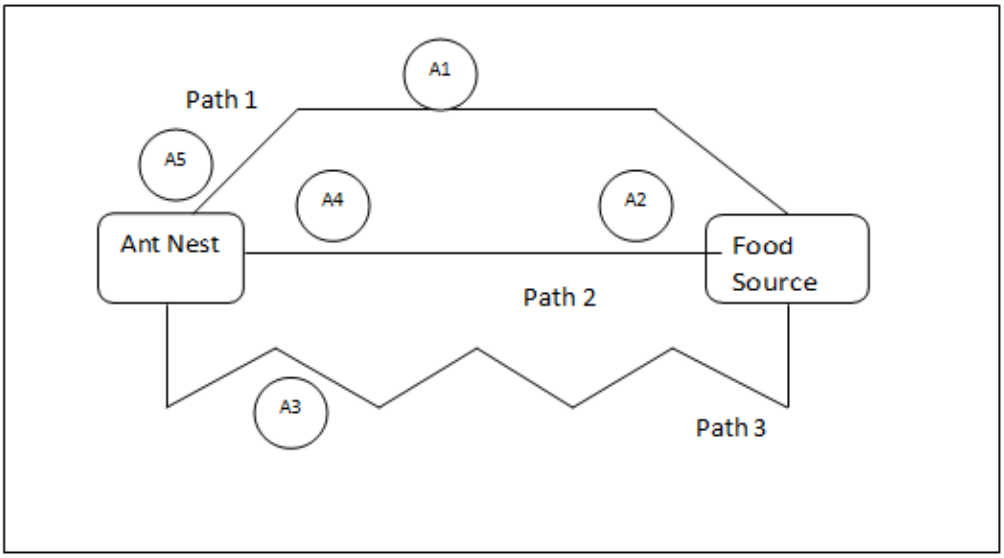

Figure 3.2. More ants deposit more pheromone on different paths

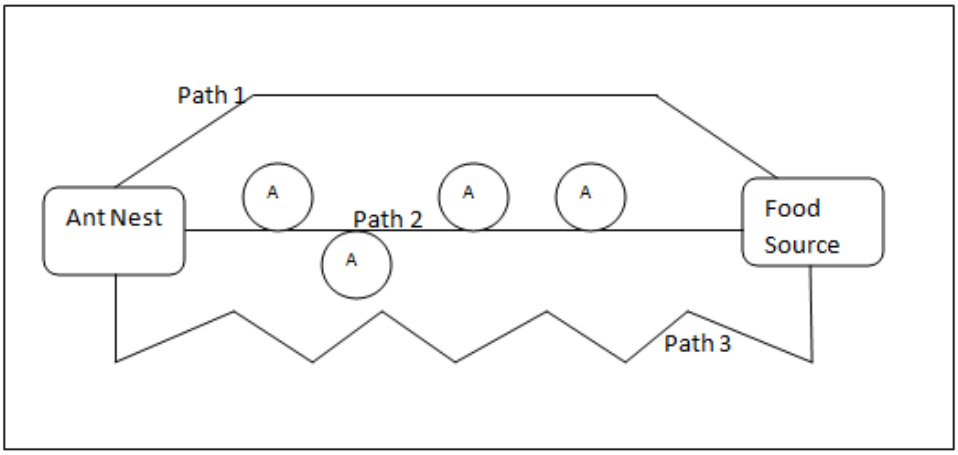

Figure 3.3.Finally all ant coverage to follow shortest path

IV. TEST CASE PRIORITIZATION USING ANT COLONY OPTIMIZATION

An extensive literature review was carried out according to the guidelines laid by kitchenham [9]. Some of the work has been discussed as follows.

Suri, B. and Singhal, S. used time aware test case prioritization technique in order to reorder the test cases of regression suite [24]. Test cases were prioritized in such a way that maximum number of faults be detected in minimum execution time. The work implemented an ant colony optimization algorithm for test case selection and prioritization. The work presented a tool ACO-TCSP implemented in C++ to solve Test case prioritization problem. ACO-TCSP model was used in order to perform test case prioritization within time constraint environment. Inputs to the system were fault detection information and execution time information of the test suite along with the time constraint detail. Test cases were prioritized on the basis of total fault detection and minimum cost of execution. The output generated included path details for each iteration, pheromone details, best path details and the final selected and prioritized test suite.

The proposed algorithm was coded as ACO-TCSP in C++. Algorithm consisted of 10 modules having 5 global functions, 13 global variables, 2 stuctures and one class.Modules implemented were init_ant(), Enter_ts(), newnode(int), ACO() and main(). ACO-TCSP was run for four times on the example proposed by singh [2010, acm]. ACO-TCSP resulted in 62.5% reduction in test suite size of regression testing.

In another work, Bharti Suri and Shweta Singhal, anaysed ACO algorithm for test case selection and prioritization and conducted that proposed algorithm finds better results at high values of time constraints [25]. The work utilized ACO in order to solve time constrained prioritization problem for regression testing. The work was implemented on 7 C++ and one java program named collAdmission, HotelMgmt, triangle, quadratic, cost_of_pub, calculator, prcv_day and railway_took.(check)

These selected programs varied from 31 LOC to 666LOC. The details of number of versions, no. of test cases and execution time of these programs are specified in []. 10 runs of proposed ACO algorithm was executed for each of the programs with seven different time constraint values. The algorithm was executed 70 times with varying time constraint values as 85, 150, 200, 250, 300 350 and 400.

The work observed that number of optimum paths increased with the increase in time constraint value. Best paths were found according to total fault coverage criteria in minimum execution time. The work concluded that higher value of time constraints leads to more optimum paths and more reduction in execution time.

In another work applied ACO approach was applied to select and prioritize regression test cases in a time constrained environment [26]. Experiment was conducted to evaluate the effectiveness of proposed algorithm. The work also compared the proposed technique with other by using APFD metric. The proposed algorithm was implemented on 2 C++ programs having 281 LOC and 666 LOC. After 40 test runs of ACO algorithm on the program percentage of reduction in execution time was 82.5% and 58.17% respectively. The work was compared with following techniques using 2 C++ programs-

- No order
- Random order
- Reverse order
- Optimal order
- ACO order

Comparisons were made using APFD metric. Results showed that ACO technique leads to optimum ordering.

Tadohiro Noguchi et. al. [27] proposed the framework to prioritize test cases for black box testing. Test cases were prioritized on the basis of test execution history of similar product. The work proposed the use of ant colony optimization to prioritize the test case categories. The work was evaluated using APFD (Avg percentage of fault detected). The work was implemented on a medical software (as prior product) (17000 test cases, 100 test categories, 1400 faults) and financial software (as target product)(3000 test cases, 50 test categories ,80 faults). Average APFD generated was 0.94 and time to finish prioritization was 9.3 seconds.

Gao. D. et. al. proposed an algorithm to prioritize test cases based on ACO[28]. The work considered three factors i.e. number of faults detected, execution time and fault severity. The effectiveness of the algorithm was demonstrated using APFD metric. The results showed optimized ordering of test cases.

V. CONCLUSION

ACO is a metaheuristic search technique used to solve optimization problem. A large number of optimization problem have already taken advantage of ACO technique while countless other are on their way. Test case prioritization calls for re-ordering of test cases based on some criteria such that important test cases are executed well in time. The paper presents various test case prioritization techniques based on ACO. The work concludes that ACO technique leads to optimum ordering of test cases.

References

[1] Bertolino, A. (2005). A brief essay on Software Testing (Vol. 1). Wiley-IEEE.
[2] Myers, J. 2004. The Art of Software Testing (second edition ed.). John Wiley and Sons.
[3] IEEE std. definition of Regression Testing.
[4] Bhasin, H. et. al. 2015.Neural Networks based Automated Priority Assigner. Series Advances in Intelligent Systems and Computing. Springer. Volume 381.
[5] Klir, G. J., Yaun, B. 1995. Fuzzy sets and Fuzzy logic theory and applications. Appendix B- Genetic Algorithms- An overview. Pearson Education Inc
[6] Khanna, E. 2016. Regression Testing based on Genetic Algorithms. International Journal of Computer Applications
[7] 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.
[8] Bhasin, H., Manoj. 2012. Regression Testing Using Coupling and Genetic Algorithms. International Journal of Computer Science and Information Technologies. Vol. 3(1).
[9] 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
[10] Chauhan, N. 2010. Software Testing principles and practices. Oxford University Press.
[**11]** Srivastava, P., R. 2008. Test Case Prioritization. Journal of Theoretical and Applied Information Technology.pp 178-181.
[12] 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.
[13] 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
[14] 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
[15] 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.
[16] 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.
[17] Dorigo. M, et. al. 1996. Ant System: Optimization by colony of operating agents. IEEE transaction on system, man and cybernetics. Vol B(26), 29-41.
[18] H.Li, and C.Peng Lam. 2005. "Software Test Data Generation Using Ant Colony Optimization," In Transactions on Engineering, Computing and Technology.
[19] R.S.Parpinelli, H.S.Lopes, and A.A.Freitas,. 2002. Data mining with an ant colony optimization algorithm. IEEE Transactions on Evolutionary Computation, vol. 6, pp. 321–332.
[20] P.Zhao, P.Zhao, and X.Zhang. 2006. New Ant Colony Optimization for the Knapsack Problem. Proceedings of the 7th International Conference on Computer-Aided Industrial Design and Conceptual Design. pp 1-3.
[21] K. Doerner, W.J. Gutjahr. 2003. Extracting Test Sequences from a Markov Software Usage Model by ACO. In the Proceedings of GECCO, LNCS Springer-Verlag Berlin Heidelberg, Vol. 2724,2003, pp. 2465-2476.
[22] F P. McMinn, M. Holcombe, M. 2003. The State Problem for Evolutionary Testing. In the Proceedings of GECCO 2003, LNCS, Springer Verlag, Vol. 2724, 2003, pp. 2488-2500.
[23] Suri, B., Singhal, S. 2012. Literature Survey of Ant Colony Optimization in Software Testing.
[24] Suri, B., Singhal, S. 2011. Implementing Ant Colony Optimization for test case selection and prioritization. IJCSE. Vol 3(5).
[25] Suri, B., Singhal, S. 2011. Analysing test case selection and prioritization using ACO. ACM sigsoft software engineering notes. Vol 36(6). 1-5.
[26] Suri, B., Singhal, S. 2011. Test case selection and prioritization using ant colony optimization. International Journal of computer science and its application.
[27] Tadohiro Noguchi et. al. 2015. History based test case prioritization for BBT using ACO.
[28] Gao. D. et. al. 2015. Test case prioritization for regression testing based on ant colony optimization. IEEE.