2020 8th International Conference on Reliability, Infocom Technologies and Optimization (Trends and Future Directions) (ICRITO)
Amity University, Noida, India. June 4-5, 2020

SARLA - A 3-Tier Architectural Framework Based on the Aco for the Probablistic Analysis of the Regression Test Case Selection and their Prioritization

Prashant Vats
Research Scholar, AIMACT,
Banasthali Vidyapith, Rajasthan, India
prashantvats12345@gmail.com

Anjana Gossain
USICT, G.G.S.I.P. University
New Delhi, India
anjana_Gosain@yahoo.com

Manju Mandot
Janardan Rai Nagar Rajasthan Vidyapeeth,
Udaipur, Rajasthan, India.
manju.mandot@gmail.com

***Abstract*—Software testing is a process of testing Software under Test (SUT) with the intent of finding errors in it & plays a major role in the software development process. Further the Maintenance phase of any software product needed to go through a series of Regression testing process. In Regression Testing, It is required to retest the existing software module whenever any modification is done to that module in order to see & check its functioning after the removal of errors in it. During the Regression Testing it is a necessary habit that we use to perform certain steps like Test case selection, their effective prioritization. To select and chose an effective set of prioritized test cases that ensures that all the faults are being covered quickly with the minimum execution time . So in other words, we can say that Regression test selection is a process of reducing the number of test suites by selecting the appropriate subsets from an original test suite to ensure the hundred percent code coverage of a SUT. In this paper, we have proposed a 3-tier Architectural framework called SARLA that provides us a cost effective method for the probabilistic analysis of the Regression test case selection and their prioritization based upon the Ant Colony Optimization (ACO) technique.**

Keywords- Regression Testing, Test case selection, Test suite prioritization, ACO.

I. INTRODUTION

The Software maintenance phase of any software during a Software Development Life Cycle (SDLC), is a very crucial process & requires the Software to go through a detailed procedure and examination involving the process of Regression Testing [10]. During Regression Testing of the SUT, it is required to again test the existing modules of a SUT to see & check that whether the correction made to that module of the SUT after the removal of errors has not introduced further a new kind of malfunctioning in the Current SUT. So, the Regression testing is done in order to validate our changes that we have made to resolve bugs in the SUT. During the regression testing, we use the existing test suites and schedule those test cases using a prior test case prioritization algorithm. In case of our proposed framework, here we have used the ACO for the test case prioritization, thus provides a minimal set of test cases using which it ensures the hundred percent code coverage of a SUT. All faults have been covered in these prioritized test suites. In this test case selection and prioritization, we have used ACO that is an optimization technique which is based upon finding an iconic solution to the given problem of algorithm by providing an optimized path as a solution to ensure the full code coverage using minimum number of efforts during the regression testing of the SUT.

II. ANT COLONY OPTIMIZATION

During the study of Entomology for the Ants, the Entomologists have made a remarkable discovery in the area of ACO[11]. They have explained how the Ants communicate when they move out of their colony in search of food & using a natural phenomenon how they trace back to their zero location using the minimum displacements from their origin thus providing a shortest path. They explained that Ants while moving out from their colony in search of food uses a chemical substance called the Pheromone. While moving onto a path In search of food, as they move around ahead onto a path they left behind a trail of the Pheromone substance on that path, which is being used by the other ants to follow that path. The residual ant when moves around onto a path they chooses a path with maximum amount of trail of the Pheromone on it, using which they not only reaches their food source but also has causes back to the location where their colony is situated. So this, it provides them a shortest path to return towards their nest which they uses as the optimized shortest path to return & spreading more pheromone onto that path while using that path a number of times [12].

III. RELATED WORK ON ACO INTO SOFTWARE TESTING

Bharti Suri, et.al, [1] has proposed their work onto Regression testing, thus proposing their work onto ACO technique as an application technique to solve the test case selection & prioritization problem in order to provide solution for the Regression Testing of SUT. They have used C++ to implement their proposed algorithm.

Neha Sethi, et.al, [2] has proposed their work again onto regression testing, thus using ACO to solve the test case selection & prioritization without using number of links that may cover extra comment lines.

Minje Yi, et.al, [3] has proposed an ACO algorithm for testing a SUT using path oriented testing by combining the ACO and the Genetic Algorithm (GA) for the generation of

978-1-7281-7016-9/20/$31.00 ©2020 IEEE

path oriented test case data. The said work is used to solve a Triangle Discrimination Problem.

Chengying M., et.al, [4] has used the ACO for generating the improved quality testing code that would be used for the purpose of earlier fault detection in the SUT. Their results have explained the ACO algorithm by using simulated annealing of the test case selected

Praveen Ranjan S., et.al, [5] has proposed a new test case prioritization algorithm which uses the Average Percentage Fault Detection (APFD) metric to calculate average faults & effectiveness of the proposed algorithm. In their work they have analyzed the SUT using both prioritized and non-prioritized test cases with the help of the APFD metric & have proved that their prioritized test cases are more effective than the conventional one.

Ruchika M., et.al, [6] has proposed a technique based upon ACO that provides version specific test cases prioritization, in which the changes in the programs are known. The proposed technique makes use of two algorithms that's modification & deletion. The proposed algorithm identifies the set of test cases that executes the modified LOC exactly ones & executes the LOC after the modification is made to the source test code & thus provides the reduced cost of regression testing of SUT.

Ashima Singh, et.al, [7] has used the Swarm Intelligence by using the Genetic Algorithms (GA) in order to provide the optimized solutions in form of optimized test cases for a SUT. In their propose work they have provided solution by sequencing the test cases & thus reducing them by using an intelligent dynamic methodology for generating the test cases based onto priorities. In their proposed work they have used a cumulative mutation probability metric approach for determining the effectiveness of the ordering of the new test cases.

A. Pravin, et.al, [8] has proposed their work based onto ACO to prioritize the test cases based on the code coverage & then improving the test process by finding the faults in an SUT on earlier basis. In their proposed work, for every test case they have generated a value depending upon the certain criteria & thus comparing them & then a priority is assigned to them based upon their individual value.

Pradipta Kumar, et.al, [9] has proposed a new test case prioritization algorithm based on GA. In their proposed work, they have separated the teat cases that were detected by the customer as severe & among the rest test cases they have prioritized the subsequences of the original test suite so that the new test suite which has to run, they can run under a time constrained execution environment. They have analyzed the GA w.r.t. the effectiveness & time overhead by using the Structure based criteria to prioritize the test cases.

IV. ANT COLONY OPTIMIZATION ALGORITHM FOR REGRESSION TEST CASE SELECTION, MINIMIZATION & PRIORITIZATION

For a Given SUT, Let T be the test cases, where T consists of a set of n number of test cases such that

T=[t1, t2, t3, t4, t5,… , tn]

Let Fault F be the faults that are existing in a given SUT, such that F= [f1, f2, f3, f4,… , Fn]

Some of the Faults among a given set of Faults F is already being covered by each test cases [T1, T2,… ,Tn]

Now, for the evaluation of a complete path followed by the Ants must be taken up to its maximum limit such that Total Time Traveled Constraint (T.T.) is equal to maximum.

Let S-> be the number of artificial ants generated, where S is equal to the number of test cases.

The proposed ACO works as follows:

Start

Initialize by putting details about the Test suites Where Ti= Original test suite where i<- 1to n

n = number of actual test cases.

Generate M number of ants for every test case such that Sj=number of ants generated for every test case

M= number of ants

Given the details about Faults Fg where g will range from 1 to n

i.e. for every test case it uncovers a fault.

Let Wi be the amount of Pheromone sprayed by the ants while every trails is covered by covering each edge between two nodes.

Given the Evaporation rate of Pheromone while every trail is covered by an ant between the nodes is = 10%

Run the ACO algorithm for all the ants Sj such that ensuring 100% code coverage.

Choose the best minimum shortest path from each iteration which is being used on the following:

Maximum possible fault coverage with minimum execution time.

Ensuring 100% code coverage.

Calculate the Pheromone to be deposited using the following formula"

$$t_{ab}^{x} = (1 - \partial)t_{ab}^{x} + \Delta t_{ab}^{x}$$

Where,

t_{ab}^{x} =amount of Pheromone to be sprayed onto a path under transition from initial state a to the final state b.

∂ = Pheromone decay rate.

Δt_{ab}^{x} =Total amount of Pheromone sprayed by the ant onto a specific transition path from a to b.

Update the Pheromone by depositing more pheromone

onto the shortest path chosen by the ants for their colony to the food source, i.e. the path chosen provides maximum fault coverage for a SUT using minimum number of test cases.

Thus it will result into reduction of pheromone onto the other edged due to evaporation or due to decay rate at 10%.

When all faults are covered in using minimum number of test cases ensuring the maximum fault coverage terminates the search or iteration process.

Check all the test cases by determine final path that is based upon the minimum execution time & maximum fault detection.

V. PROBABILISTIC ANALYSIS FOR THE RESULTS OBTAINED FOR THE REGRESSION TESTING OF A SUT USING ACO

Let the test cases selected to be I and probability of finding error with test case j is

$$Pj = \frac{[[Errors\ covered\ j]^{\beta} * [Pheromone\ train\ between\ i < -j]^{\alpha}]}{\sum_{k=1}^{n}[[Errors\ covered]^{\beta} * [Pheromone\ trail\ from\ i < -k]^{\alpha}]}$$

Where j, k € {Non – visited Test cases} This formula can be further formulated as Probability

$$\text{Probability Pj} = \frac{[Errors\ covered\ j/Execution\ time]^{\beta} * [Pheromone\ Trail]^{\alpha}}{\sum_{k=1}^{n}[\left[\frac{Errors\ covered\ j}{Execution\ time\ k}\right]\beta * [Pheromone\ Trail, k] * x^2]}$$

Where j, k {non visited test cases}

To select a set of new test cases while choosing the set of next test cases using the above formula would be used, the highest probability test case may be chosen.

VI. COMPONENTS OF THE 3 TIER ARCHITECTURAL FRAMEWORK SARLA

The given proposed framework SARLA with is components have been shown in fig. 1. The proposed 3tier Architectural framework SARLA for the regression testing of an SUT which is based on ACO consists of the following components.

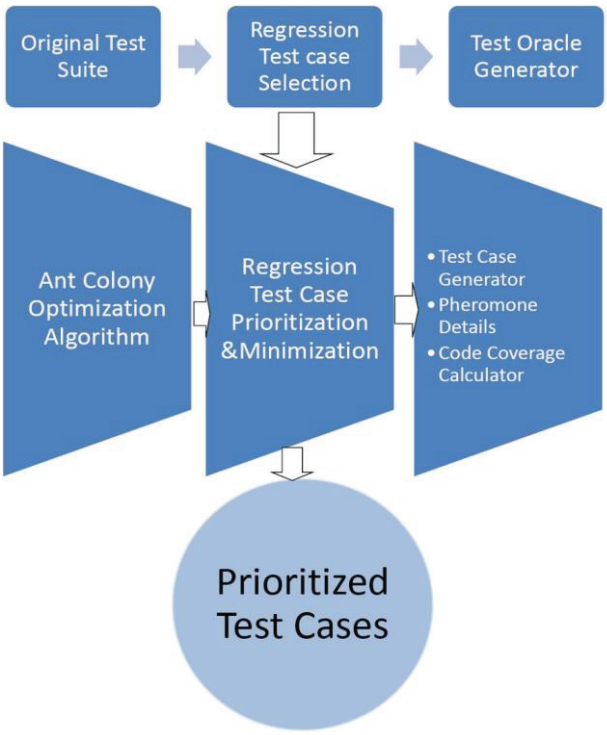

Fig. 1. To show the Concept of the 3-tier architectural framework SARLA for the registration testing using

1. Regression Test Case Selection - In this module, the original test cases are fed as an input to the to the regression test suite selection block, where Ti = Original test suite. Let m be the number of total actual test cases covering each individual fault Fi to n. It generates a set of Regression test suite "S" which is equal to the n number of ants generated for each test case. It also provides a test oracle for the probabilistic analysis of the testing of the SUT using ACO.
2. Regression Test Case Prioritization & Minimization – In this module an ACO technique is used to select, minimizing, & prioritizing them. It also ensures the putting on weight wi to each code branch covered between two nodes & if a path is not visited then it also calculates the decay time. It provides a Probabilistic analysis of the given formula for test cases which have the highest probability according to the given formula are being chosen.
3. Both the number of Faults covered in a well execution time for each test case to be considered. The results obtained have shown that the total execution time of all selected test cases from the prioritized test suite is less as compared to the result obtained from the previous formula. We can implement the algorithm & code using Java.
4. At the 3rd level of the proposed framework, various components like Pheromone details provided the test cases covered during the test paths get executed. The Test case Generator provides set of test cases that has been obtained by adding wi weight every time, whenever that particular code is revisited by "n" number of Ants. The Code coverage calculator provides whether a code path is executed or not.

The block diagram of 3-tier architectural framework SARLA is given in Fig.1.

VII. FLOWCHART FOR THE 3 TIER ARCHITECTURAL FRAMEWORK SARLA

It shows the process chart of how the ACO works test case prioritization & selection works for the 3 tier regression testing framework SARLA. It initially reads the information from the given set of the test cases that describes the number of path selections & iterations, their execution time & faults covered by the test cases. After that ACO is applied to select the test cases randomly that ensures 100% code coverage & maximum fault detection. Choose a test case that covers the entire fault then include it in the optimized solution. If all faults are not covered, then select another test suite by continuing do so until all the test cases that have the highest probability of detecting faults using ACO for a SUT are being covered. Every time with inclusion of a test case, update the pheromone amount at each edge in the pheromone table & store the execution time of all the selected test suites. Explore all the paths & the best path among all the explored path is chosen that covers the all faults with the minimum execution time. Different test cases for the faults with the minimum execution time. Different test cases for the test suite are selected & then prioritized according to the modified probability formula and then results in reduced cost of the regression testing for a SUT. The said flowchart is given in Fig. 2.

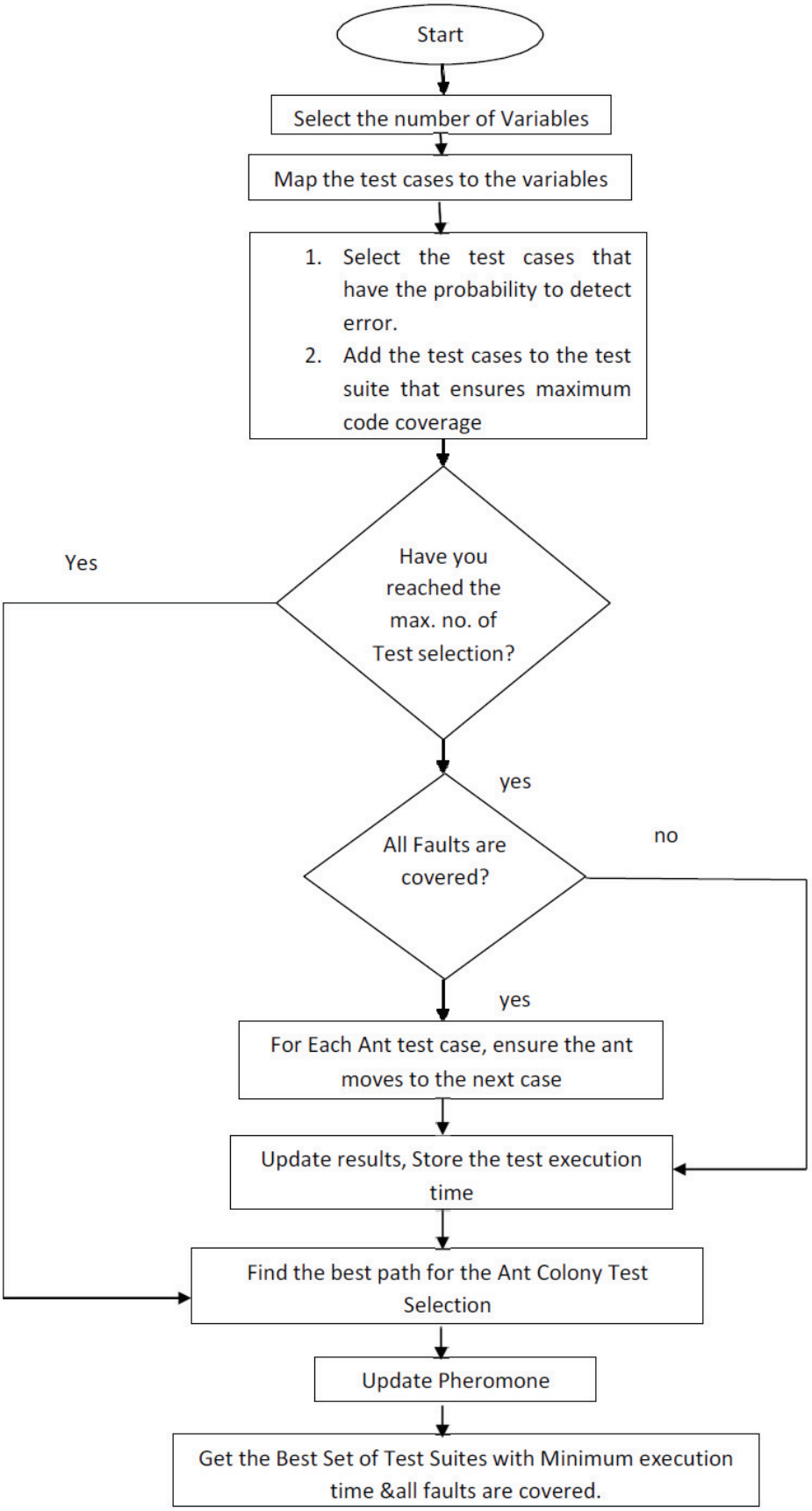

Fig. 2. Flowchart for showing the working of the ACO technique in the proposed framework SARLA

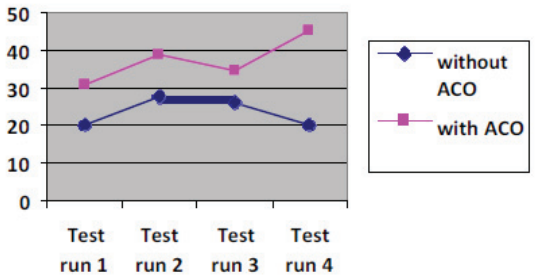

Fig. 3. The probability of fault coverage using ACO and without ACO

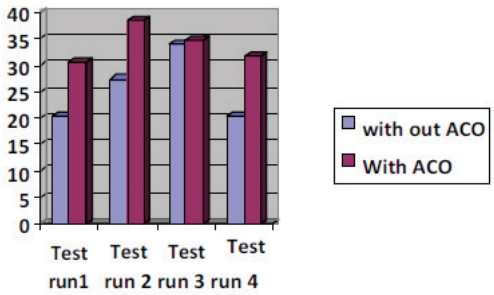

Fig. 4. The test results running with ACO and without ACO reference

VIII. EXPERIMENTAL SETUPS & RESULTS

For the implementation of the proposed framework we can use java for the implementation. For the implementation of our framework we can use the following modules.

1. Class Init Ant

{

Void init ANT();

}

2. Class New Node

{

New Node N= new New Node();

}

3. Class ACO

{

ACO Ant1= new ACO();

}

4. Class Regression Test Suite

{

Public Static Void Main (String args[])

{

}

}

Apart from this we have used an Input table for 7 test case & 10 faults, is given with their execution time in Table 1. Subsequently Test Results are shown in Table 2, 3, 4, 5 respectively after running the Sample Runs 1, 2, 3 & 4. From our analysis the proposed method has been compared with traditional ACO for the test case selection & prioritization. By after adopting the same parameters & putting that in our prioritizing & selection algorithm we have found that our proposed approach has provided better results with improved probability for covering the left out test cases that remain untraced during the code coverage of fault detection using ACO for regression testing. The fig. 3. show the probability of fault coverage using ACO and without ACO. The fig 4. shows the test results with running with ACO and without ACO. Also, the proposed framework has provided that execution time of selected test cases has been reduced as compared to the previous one after applying the modified probability formula as compared to the previous one.

TABLE I. INPUT TABLE FOR 7 TEST CASE & 10 FAULTS, IS GIVEN WITH THEIR EXECUTION TIME.

Faults -> /Test Cases	F1	F2	F3	F4	F5	F6	F7	F8	F9	F10	NUMBER OF FAULTS COVERED	EXECUTION TIME IN MINUTES
T1	X		X		X		X		X		5	7
T2	X	X		X	X		X	X	X		7	5
T3	X	X		X		X		X		X	6	6
T4	X		X		X	X	X	X	X		6	5
T5	X		X	X					X	X	6	5
T6	X	X			X	X					4	5
T7	X	X				X				X	4	4

TABLE II. RESULTS AFTER SAMPLE RUN 1.

	Iteration	Ant	Best Path	Execution time	Result
Run1	1	A1	1,5,7,3	18	Weight as edges after all iterations 1,5: 0.54 3,4:6.13 3,7:1.13 4,5: 6.13 5,7: 0.54 Rest all has 0 weights. Execution time for all iterations is 84 Best Path found is 3,4,5 Provides Optimized Solution
	2	A7	7,3,4,5	15	
	3	A5	3,4,5	13	
	4	A3	3,4,5,6	14	
	5	A4	1,2,4,6	17	
		A5	3,4,5	13	
		A3	3,4,5,6	14	
	6	A2	1,2,4	12	
		A4	1,2,4,6	17	

TABLE III. RESULTS AFTER SAMPLE RUN 2

	Iteration	Ant	Best Path	Execution time	Result
Run2	1	A5	5,1,3	15	Weight on edges after all iterations 1,3: 7.80 1,5:7.80 Rests all have 0 weights. Execution time for all iterations is 94 Best Path found is 3,1,4
	2	A3	3,1,4	13	
	3	A4	2,3,4	14	
		A3	3,1,4	13	
		A4	2,3,4	14	
	4	A2	1,2	12	
		A4	2,3,4	14	
		A3	3,1,4	13	
		A4	2,3,4	14	
		A4	2,3,4	14	
		A5	5,1,3	15	

TABLE IV. RESULTS AFTER SAMPLE RUN 3

	Iteration	Ant	Best Path	Execution time	Result
Run3	1	A7	7,1,3,5	18	Weight on edges after all iterations 1,3: 7.80 1,5:7.80 Rests all have 0 weights. Execution time for all iterations is 82 Best Path found is 1,3,4 Provides Optimized Solution
	2	A1	1,3,4	16	
	3	A5	5,2,1	14	
	4	A4	4,5,3	15	
	5	A2	1,2	12	
		A4	4,5,3	15	
		A2	1,2	12	
		A4	4,5,3	15	
	6	A3	3,5,4	10	
		A4	4,5,3	15	
	8	A6	2,3,6	11	

TABLE V. RESULTS AFTER SAMPLE RUN 4

	Iteration	Ant	Best Path	Execution time	Result
Run4	1	A1	1,2,,3,5	16	Weight on edges after all iterations 1,3:3.24 3,4: 7.27 4,5: 4.24 4,7: 0.72 Rests all have 0 weights. Execution time for all iterations is 87 Best Path found is 3,5,4
		A1	1,2,3,5	16	
	2	A4	4, 3, 1	14	
		A1	1,2,3,5	16	
		A1	1,2,3,5	16	
		A4	4, 3, 1	14	
	3	A7	7,4,5,3	12	
	4	A2	2,5,4	13	
		A4	4, 3, 1	14	
		A4	4, 3, 1	14	
	5	A3	3,5,4	13	
		A4	4, 3, 1	14	

IX. CONCLUSION & FUTURE WORK.

In this paper, we have presented a workable 3 tier architecture SARLA that is using the ACO technique for the Regression test suite selection & prioritization. After carrying out our experimental setup, study & implementation we have observed that the use of ACO in regression test suite selection & prioritization has provided us much improvised results as compared to other approaches. Further, it has been also provided us an efficient code coverage. As further for future work the proposed framework can be implemented as an automated tool to find its applications onto large scale systems.

REFERENCES:

[1] Bharti Suri,et.al., "TEST CASE SELECTION & PRIORITIZATION USING ANT COLONY OPTIMIZATION", pub. in proceedings of International conference on Advanced Computing, Communication and Networks, 2011.

[2] Neha Sethi, et.al, "Ants Optimization for Minimal Test Case Selection and Prioritization as to reduce the Cost of Regression Testing", pub. in International Journal of Computer Applications (0975 – 8887),Volume 100 – No.17, August 2014.

[3] Minje Yi, et.al, "The Research of path-oriented test data generation based on a mixed ant colony system algorithm and genetic algorithm", pub. in proceedings of International Conference on Wireless Communications, Networking and Mobile Computing (WiCOM), Shanghai, pp.1-4, 2012.

[4] Chengying M.,et.al, "Generating Test Data for Structural Testing Based on Ant Colony Optimization", pub. in proc. of 12th International Conference on Quality Software, Xi'an, Shaanxi, pp. 98 – 101, 2012.

[5] Praveen Ranjan S., et.al, "Test Case Prioritization", pub. in Journal of Theoretical & Applied Information Technology, pp. 178-181, 2008.

[6] Ruchika M., et.al., "A Regression Test Selection and Prioritization Technique",pub. in Journal of Information Processing Systems, vol.6, pp. 235-252, 2010.

[7] Ashima Singh, et.al, "Prioritizing Test Cases in Regression Testing using Fault Based Analysis", pub. in International Journal of Computer Science, vol. 9, Issue 6, pp. 414-420, 2012.

[8] A. Pravin, et.al, "An Efficient Algorithm for reducing the test cases which is used for performing regression testing", pub. in proc. of 2nd

International Conference on Computational Techniques and Artificial Intelligence, Dubai (UAE), pp. 194-197, 2013.

[9] Pradipta Kumar, et.al,"“Analysis of Test Case Prioritization in Regression Testing using Genetic Algorithm”, pub. in International Journal of Computer Applications, vol. 75, pp.1-10, 2013.

[10] K.K. Aggarwal & Yogesh Singh, “Software Engineering”, New Age International, 2001

[11] T. H. Cormen, C. E. Leiserson, R. L. Rivest, Clifford Stein, “Introduction to Algorithms”, 3rd Ed., PHI, 2013.