Application of Ant Colony Algorithm in Regression Testing of Web Applications –A novel approach

MUNISH KHANNA
Hindustan College of Science & Technology
Farah, INDIA
munishkhanna.official@rocketmail.com

Abstract: - Regression testing is one of the most arduous and challenging activity which has been implemented during maintenance phase of the software development life cycle. With the ever increasing demand of updating the web applications to meet the never ending user's expectations and to survive in highly competitive environment gives rise to inspiration of regression testing of web application. But due to a range of constraints like monetary issues, short span of time and availability of skilled human resources; it becomes very much impractical for comprehensive regression testing of web application. In this proposed approach the user sessions which are traced out from logs of the server act as an input to the system .With the help of Ant Colony Optimization algorithm (ACO) the proposed approach will generate minimum number of test cases fulfilling certain predefined objectives. Moreover in the proposed approach, with the help of ACO and user sessions, minimum number of test cases will be generated for regression testing of the web application in hand.

Key-Words: - Ant Colony Algorithm, User Session inspired web application testing, Test case generation, Test case Optimization, Web Application Testing, Test case Reduction and Regression Testing.

1 Introduction

With the easiness in reach ability of internet, the numbers of users on the internet are increasing day by day. To satisfy the online purchasing, selling and searching demand of these users new web applications (dynamic websites) inculcating the new innovative ideas are deployed over the internet at a rapid speed. These web applications have been very much successful in penetrating customer community by providing uninterrupted, trustworthy and eminence services. This is the ultimate responsibility of the development and tester community to provide quality assurance of these web applications. The most critical issue of supervising quality assurance of the software will be only achieved when thorough testing will be done and also to verify whether the customer requirements have been fully incorporated or not. Moreover under the umbrella of software quality assurance parameters like correctness, completeness, high consistency, scalability, robustness of the deliverable software will be sort out. Out of total software development cost 50% is generally spent during the testing phase. It is practically impossible to do exhaustive testing as it leads to huge number of test cases. This gives the motivation for the development of technique which reduces the number of test cases so that tester could test only these numbers of test cases instead of all possible test cases to meet out certain predefined objectives. In this proposed work u ser session behavior is considered as a test case data covering one of the test case and is used as input to the system and then ACO is used to find set of minimal test cases from every possible test case. Blending of white box testing which takes care of structural testing and black box t esting which focus on functional testing gives rise to concept of gray box testing which incorporated the features of both of these testing. The proposed work uses gray box testing for testing dynamic websites.

The structure of a websites is analogous with the graphs in which pages will be equivalent to the nodes of the graph while links play the role of edges of the graph. The generated graph will be a connected one but not strongly connected as each page will not be connected to remaining all. Any fault on a particular page can interrupt one or more functionalities/services of the website under test..The efficacy of the testing must be calculated in terms of efficient and inclusive testing within a limited time slot without pause of services. During user session based testing field data is used to create test data. Each session describes interaction between end users and the web applications in the form of sequences of URL requests with associated name-value pairs. Creating test cases from the user sessions have been practiced by the scholars since last decade.

Elbaum et al[1] proves that the fault recognition proficiency of the test cases generated from user sessions is equivalent to white box testing of the

web application. Larger the number of user sessions the higher will be the fault detection competence.

Sampath et al.[2] applied the concept analysis for the clustering and the reduction of the user sessions and proposed 03 he uristics. The reduced set has equivalent fault detection capability as that of the original set. But managing obsolete test cases was not discussed in the proposed work.

Sampath et al.[3] prioritize the reduced test suite to increase the fault detection rate. Several heuristics were proposed to order the reduced test suite. These test suites were generated from the user sessions and were applied on the web application to seek the discrepancy between expected and actual results.

In context of web applications, regression testing comes into play whenever there is an updating, addition or deletion of the web pages/ web services/functionalities/interfaces at page level/function level/service level or at module level in the fault free previous version due to changes in user's requirement, faults and/or poor performance of the previous versions on certain factors or for providing innovative services to the customers due to high pressure of competition. . With the support of this kind of testing it must be ensured that these modifications in the existing system should not lead to any unwanted new faults. For the identification and removal of these faults, regression testing is performed within time and budget constraints. Size of the test suite increases with the introduction of new test cases for testing modifications plus the earlier ones. However some of the test cases may become obsolete due to deletion/modification activity. Increased size of test suite cannot be executed fully within given timeframe. Therefore test cases selection, test cases prioritization and test cases reduction methodologies are performed during regression testing. Earlier the identification of new test case reduction techniques and the proposal of novel test case prioritization technique were considered to be different research areas. Now research community have start talking about prioritizing the reduced test suite due to very short span of time especially in case of web applications where testers have really very short duration to even execute reduced test cases. In the proposed approach test case reduction methodology has been applied, to meet out certain predefined objectives.

ACO Algorithm and its applications

In ACO the behaviour of ants are analyzed, to understand, as how they find their food while wandering with the help of other ants. Real life ants are capable of finding the shortest path from their nest to food source by exploiting pheromone information. When ants start foraging to find their food they drop pheromone on the way and follow, in probability, previously deposited pheromone by preceding ants. The ants deposit pheromones on the ground in order to mark some favourable path that should be followed by other members of the colony. The ant moves according to the amount of pheromone deposited, the richer the pheromone trail on the path is, the higher the probability that it would be followed by other ants. Thus pheromone helps other ants to find their food. ACO technique is an algorithmic approach in which a set of artificial ants cooperate to find the solution of a given problem by exchanging information via pheromone deposited on t he edges of the graph. Set of agents called as, artificial ants, search in parallel for the solution in very large search space and cooperate each other with the help of pheromone–mediated indirect and global communication. Meanwhile the pheromone laid down by the previous ants also evaporates at constant rate. This evaporation plays key role in avoiding convergence towards local maxima/minima. ACO has been successfully applied by various researchers to find the solution of combinatorial optimization problem like symmetric and asymmetric TSP, 0/1 knapsack problem, job-shop scheduling algorithms, Vehicle routing problems, Weapon-target assignment problem, quadratic assignment problem , distributed networks and data mining.

Whenever ants start foraging they choose path based on pheromone value by given equation

$$p_{ij} \leftarrow \frac{\tau_{ij}^{\alpha}.\eta_{ij}^{\beta}}{\sum \tau_{ij}^{\alpha}.\eta_{ij}^{\beta}} \quad \text{----------(1)}$$

Here p de notes the probability to choose the path and τ denotes the pheromone value. Equation (1) is the combination of static value which is inversely proportional to the distance and dynamic value τ_{ij} i.e, pheromone and its value changes during different time periods. The density of pheromone is evaporated according to time. So evaporation takes place as-

$$\tau_{ij} \leftarrow (1-\rho).\tau_{ij} + \sum_{j=1}^{m} \Delta\tau_{ij}{}^{k} \quad \text{----------(2)}$$

As said earlier Ant choose high density pheromone path so they update the value of pheromone. So pheromone updation takes place as-

$$\tau_{ij} \leftarrow \tau_{ij} + \frac{Q}{l_i} \quad \text{----------(3)}$$

Here Q denote constant and τ_i is pheromone value.$\eta_{ij} = 1/d_{ij}$. d_{ij} is the distance between the nodes i and j.p_{ij} is the probability of selecting a path from node i to node j. α , β are parameters controls of τ_{ij} and η_{ij} .

Sharma,Srivastava et al. [4]proposed an algorithm for test sequences generation for state based testing and optimal path generation for structural testing using ACO. They proposed an algorithm with tool ESCov(Efficient software coverage)based on ACO technique to covers maximum software coverage at the cost of minimum redundancy.

Srivastava,Raghuram,Baby et al[6] proposed a model which identifies optimal/effective path(s) with the help of ACO for the purpose of structural testing in a directed graph. All of the decision nodes should be traversed at least once was the criterion of the optimality. Algorithm generates number of paths equals to the cyclomatic complexity of the code and automatically chooses that path sequence which will cover the maximum coverage criteria, atleast once.

Srivastava,Baby et al[7] in their research work compares the performance of Genetic Algorithm Vs Ant Colony optimization for transition based software testing. Their work describes the methodology for state transition based testing and its coverage level within the software. One of the prime objectives was to generate optimal and minimal test sequences automatically for the complete software coverage. Covering all the transitions at least once was of the interpretation of complete software coverage.

Above two published work suggests techniques for full path coverage or for optimal test data generation, but none of them suggested combine approaches for tackle both the problems together though they are very closely related.

Srivastava , Jose et al[5] worked on generation of optimal set of test sequences from markov chain based usage model using ACO. Factors like cost, average number of visits and criticality of the various states in the model were kept in mind. Other consideration includes trade-off between cost and optimality of the test coverage. The test cases generated with the usage of proposed technique gives priority most critical states and transitions. These most critical states have been covered irrespective of the cost limitation.

Suri et al[8] seeks the usage of ACO in reordering of the test suite in time constraint environment and analyse the behaviour on eight programs under test.

Singh et al[9] acknowledge the application of ACO to prioritize the test cases with the objective to identify maximum number of faults within given minimum time period. The problem of prioritization is formulated in terms of 0/1 knapsack. They confirm that the APFD % in case of optimal fault coverage and in ACO was equivalent.

Srivastava et al[10] extends and improved the proposed approach of [6] which was having the shortcoming of generating redundant paths and thus wastage of resources like time and effort. He proposed the algorithm of $O(n^2)$ which smartly chooses those nodes for traversal which gives rise to new independent path surely.

Suri et al[11] analyzed the effect of time constraint on ACO based technique for test suite selection and prioritizations during regression testing .One of the finding was if test cases will be ordered using proposed approach the faults will be detected earlier. The analysis was done on eight test programs.

2 Web Application Testing

Request Dependency Graph (RDG) of the website is a directed graph where a web page represents node and linking of page or request represents link between the nodes. There will be multiple possible paths from a source node (index page) to any other node of the website which the user visited last. Traversing a specific path with corresponding test data acts as an execution of a particular test case. Multiple test data can be generated and executed for testing the same path (execution of a particular test case). Assigning the weights on the edges of the graph will transform it into weighted directed graph where weight defines significance of the link and the nodes connected to it at both ends. Larger the weights significant the nodes and links are. Larger weighted links and their connected nodes cannot be ignored during the testing phase. The weights on the edges are nothing but the average of weights of the nodes at its both ends. The proposed approach will try to traverse all of the larger weighted nodes (also called as significant nodes or important nodes). As thorough testing is not possible (all the paths in a graph which are starting from home page) within given time frame, the proposed approach will try to cover almost all the nodes of the graph along with all the significant nodes, but not each and every edge. Now the objective is defined as finding the minimum number of paths (reduced number of test cases) so that all the significant nodes should be tested along with edges with larger weights .With very short span of time in hand, if tester executes

these numbers of test cases it can be supposed that maximum possible testing is done. In the paths to be tested repetitions of nodes and edges are allowed but repetition of cycle is not allowed. There are various possible valid paths in graph because there is only one source node (index.jsp) and so many destination nodes (because generally no end point is defined for any website and moreover user can close website at any instant of time or he/she may log out by clicking logout option). Significance and calculation of weights is discussed in coming part of the section.

In today's scenario highly professionally made E-commerce websites like amazon, alibaba and flipkart are selling billions of products and build up of very large number of pages. But many researchers have used one standard website of online bookstore whose source code is available online and consists of 10 pages which is very smaller then the professional ones. For applying the proposed technique and evaluating the performance of proposed approach a PHP based three websites of 10 pages 20 pages and 40 pages respectively are created.

Two matrices , first named as adjacency matrix is created to verify whether there is a link between $page_a$ and $page_b$ and the second one named as weighted matrix is used to depict the positive weight on the node joining $page_a$ and $page_b$ otherwise zero. It is assumed that 50% of the nodes (pages) of the website (now represented as w eighted graph) are significant. These nodes can be easily calculated by sorting all nodes in decreasing order of their weight. During testing phase these 50% significant nodes must be evaluated. So it is expected that the output of the proposed approach, which is nothing but test case(s), must test all significant nodes, however the locations of these nodes can be scattered throughout the graph or can be in continuity. Results of the proposed approach will depict the percentage of the significant nodes covered, total nodes covered and percentage of total edges covered with the execution of minimum number of test cases.

As the normal user visits very less proportionate of the total number of pages available in the website, the user sessions selected in the proposed approach will vary in length and covers very small portion of the actual graph of the website under test. To inculcate this feature in the proposed work, 50(10 more than the number of the pages in the website) user sessions are taken initially. These 50 sessions vary in length as follows. 25% of the user session's length will be less than or equal to 50% of the total number of the nodes in the graph. Next 25% of the user session's length will be less than or equal to 40% of the total number of the nodes in the graph. Remaining all i.e, 50% of the user session's length will be less than or equal to 25% of the total number of the nodes in the graph. Some exceptions with respect to above rule are made during the testing of 10 pages website.

Each user session (sample shown below) consists of the elements like user's ip address, session time and pages visited by users. These visited web pages acts as nodes of the weighted graph and the visited web pages in a particular user session act as a execution of a test case.

192.168.0.100 - - [03/Dec/2010:10:18:50 +0800] "GET /AdvSearch.jsp HTTP/1.1" 200 5231
192.168.0.100 - - [03/Dec/2010:10:18:55 +0800] "GET /Books.jsp?name=&author=& HTTP/1.1" 200 14882
172.19.153.224 - - [05/Dec/2010:15:14:30 +0800] "GET /Books.jsp?category_id=3&name= HTTP/1.1" 200 13084
172.19.153.224 - - [05/Dec/2010:15:15:08 +0800] "GET /AdvSearch.jsp HTTP/1.1" 200 5231

(Sample log file of jsp based website)

The weight of the edge is calculated as the average of weights of the two end nodes. However the value of each node is calculated by dependency of number of links on t he node (which is in degree plus out degree of the node) and information/entropy of that node. Weight w_{ij} of the existing link between any two nodes i and j is calculated as

$$w_{ij} = \frac{v_i + v_j}{2} \quad \text{... (4)}$$

Value of each node is calculated using equation 5 and then calculated value is normalized so that equal importance will be given to both of the factors i.e, dependency as well as entropy/information available at that node, which is obtained from user sessions

$$v_i = d_i \times e_i \quad \text{... (5)}$$

Here d_i define dependency of link on node i and e_i define entropy of node.

$$e_i = -\Sigma p_i log_b p_i \quad \text{... (6)}$$

Here p_i is probability of node/page selection and b is total number of pages.

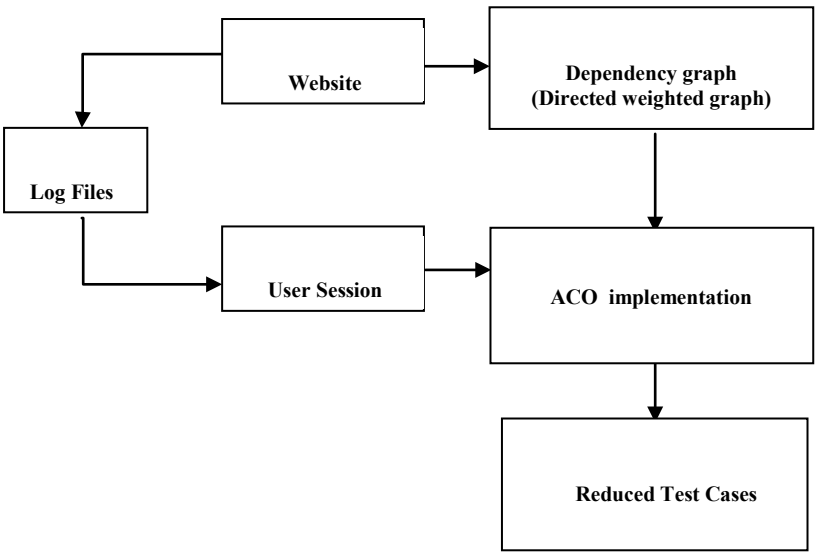


Figure F-1: Framework of Proposed Approach

In certain scenarios few significant nodes may not be visited during any of the user session and therefore isolated from testing, even being significant node. This factor supports white box testing. Moreover, larger the hit count of the particular page, the important the page (node) is and therefore it will have larger information associated with it and hence cannot be ignored during testing. This factor supports black box testing. Eventually in the proposed approach an equal importance is assigned to structure as well as behaviour. Muang and Win et al. [12] in their empirical work also uses entropy for the reduction of test cases by applying user session based approach. The proposed approach reduces the number of sessions on the basis of entropy only.

In the proposed approach sessions are used to find the entropy of the node which means that how much information is stored in the node. Higher the information stored in the node more the useful node is and therefore cannot be ignored during test case generation. Moreover this information is converted into weight of the node by joining with dependency of the node. The framework of the proposed approach is shown in the form of figure F-1.

3 Applying ACO in Web Application Testing

3.1 Minimization of Test Suite

As discussed in the previous section ACO has been effectively applied in solving numerous optimization problems of various engineering fields. In any web site there can be numerous possible test cases and it is not feasible to execute each and every possible test case due to hard deadline of delivery date of the product or even during maintenance phase where certain modifications into the existing system is to be validated. In order to find the reduced minimal number of test cases which are enough to test all possible web pages and to give confidence to the tester (by testing all significant nodes) ACO technique has been applied. Different phases of the proposed ACO are implemented as follows

Pre-processing phase

The algorithm will be iterated not more than ten times the number of pages in the graph .User sessions (Initial solutions) vary in sizes which depend upon the number of pages accessed by the user during that session, the encoding technique used is discrete values in decimal numbers. Some (in this case, 10 solutions of small size) solutions are intentionally added in the initial solution pool so that if some of the nodes (pages) are not the part of any session they may be traced out. These solutions are generated randomly with the help of roulette wheel and the adjacency matrix of the website, for path verification. Remember the best solution found so far.

Input to Algorithm::

1. Two dimensional Adjacency matrix of size 40 by 40 having value 1 if there exists link dependency or data dependency between $page_a$ and $page_b$ otherwise 0
2. Two dimensional session matrix containing all the user sessions retrieved from the log file of the web server.
3. Two dimensional weight matrix depicting the positive weights on each of the edges if connectivity exists otherwise assigned weight will be 0

Output from the Algorithm

Minimum number of test cases which will traverse through all significant nodes and almost all remaining nodes.

Since in any dynamic web site there can be many possible test cases (independent paths) and it is not possible to test every possible test case and moreover with every possible combination of test data. ACO is applied to find minimal test cases, to test the website under test, which are enough to test all possible web pages.

Request dependency graph (RDG) of the website is a directed graph where a web page represents nodes and linking of page or request represents link between nodes. So now objective is to find

minimum number of higher weighted valid paths that covers set of sub graphs or sub trees (which have highest weighted sum) of dependency graph, and the union of these covers almost whole graph (dependency graph). These sub graph or paths play the role of reduced test cases for the website testing.

For the implementation of ACO fixed number of ants has been chosen and the movement of these ants will be supported by some initial solutions in hand. Here user sessions which are retrieved from session log file can be defined as initial solutions and will guide the movement of the ants during initial phase.

03 websites are constructed for the purpose of assessment of the performance of the proposed approach, in which one is having equal number of pages as that of Online Book Store (OBS) website and remaining 02 with larger than that of previous one . The table depicts user session s_1 to s_{20} with respect to the web pages visited by the user's u_0 to u_9. Value 1 represents page is visited by the user u_i during session s_i and no value represents unvisited page.

Classical ACO technique has equation (eq-1) in which τ_i is inversely proportional to l_i because smaller the length (or weight) of link is, the higher is the pheromone value will be. In the proposed approach w_{ij} is inversely proportional to l_{ij} .

$$w_{ij} = \frac{1}{l_{ij}}$$

So eq-3 becomes:

$\tau_{ij} \leftarrow \tau_{ij} + Qw_{ij}$ ... (7)

While implementing the algorithm initial pheromone value for each link is considered to be same. The values of various parameters as follows.

$\tau_{ij} = 0.5$ $\rho = 0.01$ and values of α, β are $\alpha = 1.5$ $\beta = 1$

Equation (7) shows that pheromone deposition on the edges traversed by the ant during that path coverage depends upon total weight of the tour. Pheromone deposition on the edges of the tour will be dynamic in nature i.e, higher the total weight of the tour traversed higher the pheromone deposition will be. During the proposed work $\beta = 0$ was experimented in equation (1) which signifies that the probability of choosing the next node depends only upon the pheromone. The results were compared with $\beta = 1$ with which means probability of choosing next node depends not only on pheromone deposited on the edge but also on the weight of the edge. The results were not so promising in case of $\beta = 0$ and therefore approach was leftover.

Implementation of ACO is as follows:

After encoding of problem, ACO is applied on the website under test. Following steps are executed to implement ACO for the generation of reduced test cases of website in hand.

Step 1:

During this step initial pheromone is laid down on the edges of the nodes of the website (graph) which are directly connected to each other. These nodes and edges can be easily found out from adjacency matrix of the graph. The value of initial amount of the pheromone which is deposited on the edges is 0.5. After first and subsequent iterations value of pheromone will get changed according to equation (2). This step is executed before the starting of the iterations which means at the time of initialization of ants. During initialization phase different sessions retrieved from log file are analyzed and 3 best tours (user sessions) are chosen on the basis of fitness. Pheromones are added on the edges which are traversed during these tours as well as evaporated from the edges which are not traversed. Tour with highest fitness will deposit more pheromone then that of tour with lesser fitness.

Step2:

Actual iteration begins during this step. Number of iterations (predefined_value) will be equals to ten times the number of pages (nodes) in the website (graph) under test. Number of ants per iteration will be equal to number of pages in the website. Go to step 3.

Step 3:

The third step is to find the tour of the ants (here ant tour define as unique solution to the problem).During this step, travelling of the tour begins during the starting of the iteration. All the ants during each iteration start building their tour from the home page of the website. The path selection is done using roulette wheel technique in probabilistic fashion. At every node a roulette wheel selection function is called and this function checks the all adjacent nodes of that node and finds the probability of selecting each path using equation(1) and returns next node to be selected for path generation. In this fashion paths of all the ants will be generated. There is a probability that ant will try to move towards the edge which is having higher weights and higher pheromone value. This step will

accomplish two objectives, one acknowledged and one undeclared objective. T he acknowledged objective will be to find the longer paths in which repetition of nodes are allowed but repetitions of cycles are not allowed that is cycle should be traverse only once. The undeclared objective is to traverse the edges with higher weights.

Step 4:
During this step, pheromone updation takes place as per the equation (2). Here updation means pheromone evaporation or its deposition on the edges of the weighted graph. Pheromone evaporation is done at all of the edges while deposition is done at edges which are the part of tour traversed by ant and have highest fitness value.

Step 5:
Check for the count of iterations. If count<predefined_value then go to step 2 otherwise go to step 6.

Step 6:
This step will construct final generated solution or ants which define best solution for the problem. Among these n ants, top k ants are selected as final solution which can cover almost whole website (graph) including significant nodes.

3.2 Applying ACO in Regression testing of Web Application

Regression testing is one of the key actions performed during maintenance phase of the software development life cycle. Regression testing comes into play when alterations are performed on the website. These alterations are done due to changes in user's requirement, faults and/or poor performance of the previous versions on certain factors or for providing innovative services to the customers due to high pressure of competition. These changes can be either modifications/addition/deletion at the page level/function level/service level or at module level. The regression testing is done when the new version of the product is going to be launched with modification(s) in the previous one .Due to these modifications there is a ch ance of fault occurrence in the fault free previous version .The size of test suite increases because it comprises of all the test cases of previous version plus the new test cases for the modification part. The validation of the software after each modification is a must performing activity. There are two approaches to reduce the cost of the regression testing activity which are named as test case prioritization and test case reduction. Test case prioritization is done with the intention of finding maximum number of faults with execution of minimum number of test cases so as to maximize the factor known as A verage Percentage of Fault Detection (APFD). However various modifications have been proposed by researchers and on the basis of that different APFD are formulated with addition of various parameters like time constraint on test suite, test case execution time, severity of the faults and many more. In case of Test suite reduction, a reduced test suite is acknowledged that provides the same coverage according to some criteria as the original test suite. Earlier the identification of new test case reduction techniques and the proposal of novel test case prioritization technique were considered to be different research areas. Scenario has changed during last five years. Now people have start talking about prioritizing the reduced test suite due to very short span of time especially in case of web applications where testers have really very short duration to even execute reduced test cases. In this proposed work the reduction of test cases have been taken into consideration while performing regression testing of web application. In the proposed work two scenarios are taken into consideration in the first scenarios some of the pages have been removed from the original website .In second scenario some of the web pages are added which is equivalent to modifications in existing web pages.

3.2.1 Applying ACO in Regression testing of Web Application in deletion case

In this proposed work the reduction of test cases have been taken into consideration while performing regression testing of web application. Two scenarios are taken into consideration. In the first scenarios 03 pages have been removed from the original website while in second one 03 web pages are added which is equivalent to modifications in 03 existing web pages.

All the changes will be made in step 0 i.e, pre processing phase of 4.1 The rationalization of what extra has been done in the pre processing phase is as follows. 03 nodes have been selected for the deletion numbered as 8, 2 2 and 34. Delete all the corresponding entries from the adjacency matrix, weight matrix and entropy matrix with respect to these 03 node s. There after certain constraints have to be satisfied in the initial user sessions which are as follows.

If the deleted node comes as the last visited node , in that case the node will be directly removed from the user session .For example if 0,12,17,19,21,22 is the initial user session, then this user session will be refined to 0,12,17,19.

If the deleted nodes come as succession in the end, in this case these nodes will be directly removed from the user session .For example if 0,12,17,19,21,22,8,34,22 is the initial user session, then this user session will be refined to 0,12,17,19.

If the deleted nodes comes in between other nodes, then delete these nodes and put the node prior to the first deleted one in "**pre**" named variable and the node next to last deleted one in "**next**" variable. Now it will be verified whether there exists any direct path between pre and next . If yes then pre and next will become adjacent nodes, otherwise roulette wheel will be used to find the path between pre and next as pre acting as source station and next acting as destination station. Subsequently the user session will be updated. For example if the initial user session retrieved from web log file is 0,12,13,17,22,8,34,33,27,9 . Delete the node number 22,8,34 and assign 17 to pre and 33 to next. Now verify whether 17 and 33 are adjacent nodes or not, with the help of adjacency matrix. If yes update the user session otherwise use roulette wheel for the finding the path.

After the completion of this step all the user sessions, adjacency matrix, weight matrix, entropy matrix are updated and ready for first step of the ACO

3.2.2 Applying ACO in Regression testing of Web Application in updating case

Similarly in this case also all the changes will be made in step 0 i.e, pre-processing phase of 4.1 The good reason of what extra has been done in the pre-processing phase is as follows. 03 nodes have been added in the website (graph) and numbered as 41, 42 and 43. Add all the corresponding entries in the adjacency matrix with respect to these 03 nodes. There after certain constraints have to be satisfied in the initial user sessions which are as follows. As these are the new nodes and therefore they will not be the part of any initial user session. It must be also ensured that these nodes should be tested and cannot be left out during regression testing at any cost. For this assign the weight equal to the highest weight of the node of the previous graph. The intention of assigning highest weight is somewhat biasing ant movement towards these newly added nodes. Apply the roulette wheel for generating some initial solutions in which at least one of the newly added nodes must exist. After the completion of this step all the user sessions, adjacency matrix, weight matrix, entropy matrix are updated and ready for first step of the ACO algorithm .

4 Experimental Setup & Results

Website under test is converted into weighted directed graph where node of the graph is equivalent to a web page and link between nodes is defined as connectivity between two pages via simple link or query execution. The weight of link is calculated as average of product of entropy and degree of any two adjacent nodes i and j.

$$W_{ij} = (e_i.d_i + e_j.d_j)/2 \ldots (4)$$

In eq. (4) e_i is entropy of any node i and d_i is defined as total sum of in degree and out degree of any node i .Here entropy of any node is calculated on the basis of user sessions and depicts importance or total information of that node .Entropy is based on user sessions and calculated as per the equation given below.

$$e_i = -\sum_{n=1}^{k} p_i log_b p_i \quad \ldots (6)$$

e_{act}=75% *e_i+25%* actual time spend the page-(7)

In eq. (6) p_i is the probability of selection of any node i among all active nodes in that session. b is the total number of pages in any website and k is the total number of user session. Actually e_i is the combination of entropy and time spend on that particular page by all the users.

User's behaviour is used as one of the evidence because user behaviour has been defined through a branch 'analytics'. Here by analytics we mean to discover, interpret and communicate some meaningful data to the proposed model. User behaviour analytics plays a key role in enterprise management, marketing, risk and traffic analysis.

Here, we have use the aspect of data logging in which prefer on log analysis (system or may be network). In computer science the management of log, intelligence and log analysis is an art and science that make sense of records generated by computer (logs).

People perform log analysis:

- To compliance with security policy
- To compliance with regulation policy.
- To analyse errors.

	Entropy Calculation									
	1	2	3	4	7	5	6	8	9	10
	Default. jsp	Rgistration. jsp	AdvSearch. jsp	Login. jsp	Books. jsp	MyInfo .jsp	BookDetail .jsp	ShopRecord. jsp	ShopCart. jsp	Payment. jsp
Degree	9	3	6	10	11	10	5	5	9	6
	-0.14	-0.14	-0.14	0.00	-0.14	0.00	-0.14	0.00	0.00	0.00
	-0.14	0.00	-0.14	0.00	0.00	0.00	-0.14	0.00	-0.14	-0.14
	-0.14	0.00	0.00	-0.14	-0.14	-0.14	-0.14	0.00	0.00	0.00
	-0.15	-0.15	0.00	-0.15	0.00	-0.15	0.00	0.00	0.00	0.00
	-0.13	0.00	-0.13	-0.13	0.00	-0.13	-0.13	-0.13	0.00	0.00
	-0.13	0.00	0.00	-0.13	-0.13	-0.13	-0.13	-0.13	0.00	0.00
	-0.14	0.00	0.00	-0.14	0.00	-0.14	0.00	0.00	-0.14	-0.14
	-0.13	0.00	-0.13	-0.13	-0.13	-0.13	0.00	0.00	0.00	-0.13
	-0.14	-0.14	0.00	0.00	-0.14	0.00	-0.14	0.00	0.00	-0.14
	-0.15	-0.15	0.00	-0.15	0.00	-0.15	0.00	0.00	0.00	0.00
	-0.15	0.00	-0.15	0.00	-0.15	0.00	-0.15	0.00	0.00	0.00
	-0.16	0.00	0.00	0.00	-0.16	0.00	-0.16	0.00	0.00	0.00
	-0.14	0.00	-0.14	-0.14	-0.14	-0.14	0.00	0.00	0.00	0.00
	-0.14	0.00	-0.14	-0.14	-0.14	-0.14	0.00	0.00	0.00	0.00
	-0.13	0.00	0.00	-0.13	0.00	-0.13	0.00	-0.13	-0.13	-0.13
	-0.14	0.00	0.00	-0.14	0.00	-0.14	0.00	0.00	-0.14	-0.14
	-0.15	0.00	0.00	-0.15	0.00	-0.15	0.00	-0.15	0.00	0.00
	-0.14	-0.14	-0.14	0.00	-0.14	0.00	-0.14	0.00	0.00	0.00
	-0.13	0.00	0.00	-0.13	0.00	-0.13	0.00	-0.13	-0.13	-0.13
	-0.14	0.00	-0.14	-0.14	-0.14	0.00	0.00	-0.14	0.00	0.00
	-2.81	-0.72	-1.25	-1.94	-1.55	-1.80	-1.27	-0.81	-0.68	-0.95

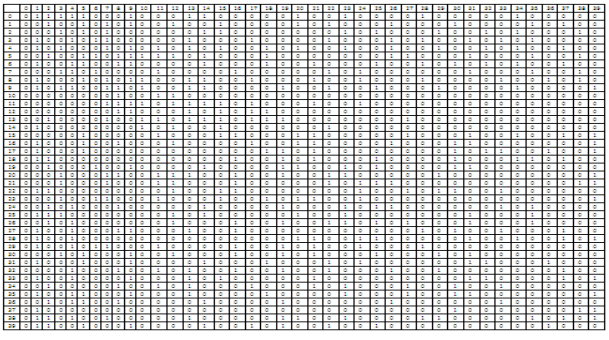
- For security incident response.

A log analysis helps in mapping of varying terminologies into normalized terminologies so that reports and statistics can be compiled together from heterogeneous environment. Thus, log analysis have their existence right from retrieval of text is to reverse engineering of software.

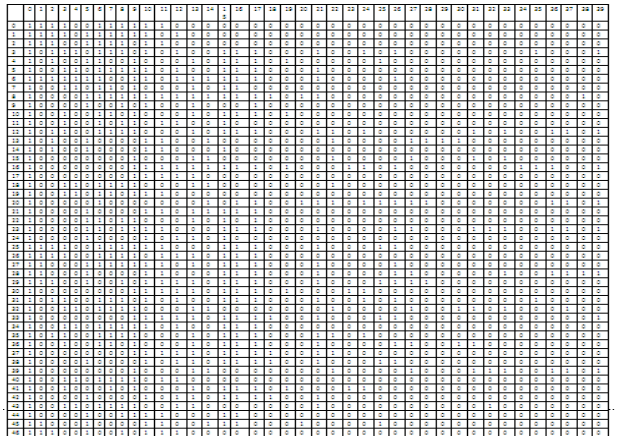
Tracking user behaviour is one of the key issues in this proposed methodology. It has been found that researchers are now raising questions on quality of user sessions by just only considering the hits (page visited) .One of the major parameter which is missing ,is the time spend on each page. Time spend on each page is assumed as directly proportional to the interest of the user and therefore should also be considered during web application testing. During background study it has been found that very less work has been published while considering both of these parameters and what percentage should be given to each of these factors. To implement this parameter various readymade tools Google Analytics, Stat Counter, Deep Log analyzer were analysed but none of them have this feature. Ultimately a script has been made in the PHP by one of the author and incorporated in the program to calculate the time spend on each page. Actually e_{act} is the combination of entropy and time spend on that particular page by all the users. For the sake of simplicity we have considered 25% weight age to time spend on each page and 75% to the entropy. Equation (7) e_{act} consists of 75% entropy of(e_i) and 25%of the time spend on that page by all the users.

Three websites (one on JSP technology and remaining two on PHP) of 10, 20 and 40 pa ges respectively are created and tested to quantify the performance of the anticipated approach. One of the main objectives of the proposed work is the identification and testing of the paths with maximum traffic with the help of entropy, which means the pages which are mostly visited by the users. The other objective is to find the path, and then test, which contains pages having larger degrees. Moreover in perspective of regression testing two modifications has been done. During the first modification three pages have been deleted from the original website, while in another modification three new pages have been inserted into the original website. All the coding implementation of proposed approach has been done in java. T hese websites were launched on the intranet of the college for one month and students of second year and final year of various branches were asked to visit the websites so as to capture the behavior of different type of the users having different maturity level. Faults related to data dependency and link dependency, in equal ratio, was seeded into the running websites in the ratio of the number of pages. Larger the number of pages in the website, larger the fault seeded. The faults were injected in the nearby area of the significant nodes. The intension of doing so is clear which is nothing but 100% testing of significant nodes and thus to achieve major objective of the work. Initial Entropy matrix of 10 pages website (EM-1), Initial Adjacency Matrix of 40 pages website (M-1),Intitial user session matrix of 40 pages website(USM-1), initial user sessions for 10 pages website(U1),initial weight matrix for 10 pa ges website (WM-1) is shown below.

EM1: Entropy matrix for 10 pages website



M-1:: Adjacency Matrix (40*40) for 40 pages



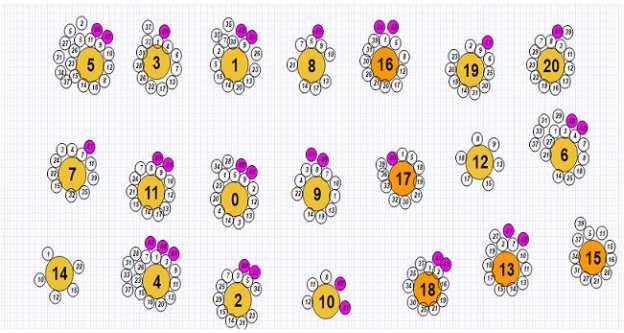
USM-1:: Initial User Session Matrix Adjacency Matrix for 40 pages

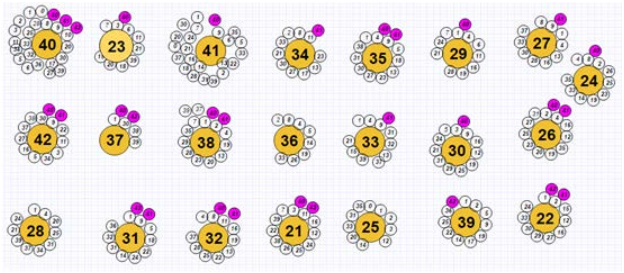
USER SESSIONS									
	1 Default.jsp	2 Rgistration.jsp	3 AdvSearch.jsp	4 Login.jsp	7 Books.jsp	5 MyInfo.jsp	6 BookDetail.jsp	8 ShopRecord.jsp	9 ShopCart.jsp
	1	1	1		1		1		
	1		1				1		1
	1			1	1	1	1		
	1	1		1		1			
	1		1	1		1	1	1	
	1			1	1	1	1	1	
	1			1		1			1
	1		1	1	1	1			
	1	1			1		1		
	1	1		1		1			
	1		1		1		1		
	1				1		1		
	1		1	1	1	1			
	1		1	1	1	1			
	1			1		1		1	1
	1			1		1			1
	1			1		1		1	
	1	1	1		1		1		
	1			1		1		1	1
	1		1	1	1				
Entropy	-2.81	-0.72	-1.25	-1.94	-1.55	-1.80	-1.27	-0.81	-0.68
Degree	9	3	6	10	11	10	5	5	9
Total node weight	25.27	2.16	7.49	19.39	17.02	17.99	6.34	4.05	6.11

U1: User Sessions for 10 pages website

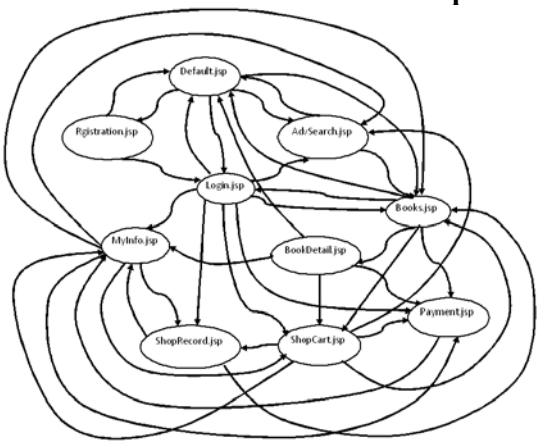
Weight Matrix										
		1	2	3	4	5	6	7	8	9	10
		25.27	2.16	7.49	19.39	17.99	6.34	17.02	4.05	6.11	5.69
1	25.27	0	13.71	16.38	22.33	0.00	0.00	21.14	0.00	0.00	0.00
2	2.16	13.71	0.00	0.00	10.77	0.00	0.00	0.00	0.00	0.00	0.00
3	7.49	16.38	0.00	0.00	0.00	0.00	0.00	12.26	0.00	0.00	0.00
4	19.39	22.33	0.00	13.44	0.00	18.69	0.00	18.21	11.72	12.75	12.54
5	17.99	0.00	0.00	12.74	0.00	0.00	0.00	17.51	11.02	12.05	11.84
6	6.34	15.80	0.00	0.00	0.00	12.16	0.00	0.00	0.00	6.22	6.01
7	17.02	21.14	0.00	0.00	18.21	0.00	11.68	0.00	0.00	11.57	11.36
8	4.05	0.00	0.00	0.00	0.00	11.02	0.00	10.53	0.00	0.00	0.00
9	6.11	0.00	0.00	6.80	0.00	12.05	0.00	11.57	5.08	0.00	5.90
10	5.69	0.00	0.00	0.00	0.00	11.84	0.00	0.00	0.00	0.00	0.00

WM1: Weight matrix for 10 pages website





Graph G1:: Graph of 40 pages website after addition of 03 p ages



Graph G2:: Graph of 10 pages website before alteration.

Graph G1 depicts 40 pages website after 03 pages numbered as 41, 42 a nd 43 have been added the website for the purpose of evaluation of regression testing. The smaller circles covering larger circles represent the outgoing from the larger ones. The considered website is heavily dense in nature that is why this way of representation is chosen. The purple colour smaller circle represents the newly added pages. Graph G2 depicts 10 pages website before any modification Partial weight matrix (PWM-1) and P artial entropy matrix (PEM-1) are depicting the part of weight matrix and entropy matrix of 40 pages website.

1	2	3	4	5	6	7	8	9	10	11	12
1.81	1.95	2.14	1.77	2.15	2.34	2.14	0.76	2.42	2.51	2.23	2.06
0.62	0.76	0.95	0.59	0.96	1.15	0.96	0.99	1.24	1.32	1.04	0.87
0.76	0.90	1.09	0.73	1.10	1.29	1.10	1.14	1.38	1.46	1.19	1.01
0.95	1.09	1.28	0.92	1.29	1.48	1.29	1.32	1.57	1.65	1.37	1.20
0.59	0.73	0.92	0.55	0.93	1.12	0.92	0.96	1.20	1.28	1.01	0.84
0.96	1.10	1.29	0.93	1.31	1.50	1.30	1.34	1.58	1.66	1.39	1.21
1.15	1.29	1.48	1.12	1.50	1.68	1.49	1.53	1.77	1.85	1.58	1.40
0.96	1.10	1.29	0.92	1.30	1.49	1.29	1.33	1.57	1.65	1.38	1.20
0.99	1.14	1.32	0.96	1.34	1.53	1.33	1.37	1.61	1.69	1.42	1.24
1.24	1.38	1.57	1.20	1.58	1.77	1.57	1.61	1.85	1.93	1.66	1.48
1.32	1.46	1.65	1.28	1.66	1.85	1.65	1.69	1.93	2.02	1.74	1.57
1.04	1.19	1.37	1.01	1.39	1.58	1.38	1.42	1.66	1.74	1.47	1.29
0.87	1.01	1.20	0.84	1.21	1.40	1.20	1.24	1.48	1.57	1.29	1.12
1.17	1.31	1.50	1.13	1.51	1.70	1.50	1.54	1.78	1.87	1.59	1.42
0.57	0.71	0.90	0.54	0.91	1.10	0.90	0.94	1.18	1.27	0.99	0.82
1.23	1.37	1.55	1.19	1.57	1.76	1.56	1.60	1.84	1.92	1.65	1.47
1.08	1.22	1.41	1.05	1.42	1.61	1.42	1.45	1.70	1.78	1.50	1.33
1.09	1.23	1.42	1.06	1.43	1.62	1.42	1.46	1.70	1.79	1.51	1.34
0.45	0.59	0.78	0.41	0.79	0.98	0.78	0.82	1.06	1.14	0.87	0.69

PWM-1:Partial Weight Matrix of 40 pages website

1	2	3	4	5	6	7	8	9	10
29.58	30.32	31.76	25.98	37.50	0.0	0.0	0.0	44.79	0.0
0.0	22.72	0.0	0.0	29.90	0.0	26.09	0.0	37.19	0.0
0.0	0.0	24.89	0.0	30.64	0.0	26.82	0.0	0.0	0.0
24.15	0.0	0.0	20.55	0.0	28.85	28.26	0.0	0.0	0.0
18.38	0.0	20.55	0.0	0.0	0.0	22.48	0.0	33.59	0.0
0.0	30.64	0.0	0.0	37.82	34.60	0.0	37.38	45.11	28.20
26.68	0.0	0.0	23.07	34.60	0.0	0.0	34.16	41.89	0.0
0.0	0.0	28.26	22.48	0.0	30.78	0.0	0.0	0.0	0.0
29.96	0.0	0.0	0.0	37.38	0.0	33.56	0.0	44.67	27.61
37.19	0.0	39.36	33.59	0.0	0.0	41.30	44.67	0.0	35.49
0.0	0.0	0.0	0.0	0.0	0.0	0.0	26.76	0.0	0.0
0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	32.44	35.82
0.0	0.0	0.0	0.0	0.0	0.0	0.0	29.10	36.38	0.0
0.0	33.01	0.0	0.0	0.0	0.0	36.38	0.0	0.0	30.57
16.99	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	15.28
0.0	0.0	0.0	0.0	39.22	0.0	0.0	0.0	0.0	0.0
27.79	0.0	0.0	0.0	35.71	0.0	0.0	35.27	0.0	0.0
25.41	0.0	0.0	0.0	33.33	0.0	0.0	0.0	0.0	0.0
13.47	14.20	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0
0.0	14.89	0.0	0.0	0.0	18.85	0.0	0.0	29.37	0.0
0.0	0.0	14.07	0.0	0.0	0.0	16.00	19.38	0.0	0.0
0.0	0.0	24.19	0.0	0.0	0.0	26.12	0.0	0.0	0.0
16.71	17.44	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0

PEM-1: Partial Entropy Matrix of 40 p ages website

Table T-1,T-2 and T-3 depicts the performance of ACO in case of reduction of the test cases and regression testing .

Characteristics	Values for 10 Pages Website	Values for 20 Pages Website	Values for 40 Pages Website
Total Number of User Sessions Considered	20	30	50
Total Number of Significant Nodes	5	10	20
Total number of nodes	20	20	40
Total number of edges	37	210	417
Total % of significant nodes covered by ACO	100%	100%	100%
Total % of nodes covered by ACO	100%	100%	97.5%
Total % of edges covered by ACO	60%	31%	27%
Total % of seeded faults covered by ACO	96%	90%	81%
Total test cases required to cover all the significant nodes, in case of ACO	2	3	5
Number of test cases required to cover all the nodes ,in case of ACO	3	4	6

T -1: Performance of ACO on reduction of test cases

Characteristics	Values for 10 Pages Website	Values for 20 Pages Website	Values for 40 Pages Website
Total Number of User Sessions Considered	20	30	50
Total Number of Significant Nodes(after addition of 3 nodes)	18	13	23
Total number of nodes(after addition of 3 nodes)	13	20	40
Total number of edges(after addition of 3 nodes)	52	243	475
Total % of significant nodes covered by ACO	100%	100%	100%
Total % of nodes covered by ACO	100%	100%	97.5%
Total % of edges covered by ACO	57%	29%	23%
Total % of seeded faults covered by ACO	92%	87%	78%
Total test cases required to cover all the significant nodes, in case of ACO	3	4	6
Number of test cases required to cover all the nodes ,in case of ACO	4	5	7

T -2: Performance of ACO when nodes are added.

Characteristics	Values for 10 Pages Website	Values for 20 Pages Website	Values for 40 Pages Website
Total Number of User Sessions Considered	20	30	50
Total Number of Significant Nodes(after deletion of 03 nodes)	04	09	18
Total number of nodes(after deletion of 03 nodes)	07	17	37

Total number of edges(after deletion of 03 nodes)	**29**	**178**	**361**
Total % of significant nodes covered by ACO	**100%**	**100%**	**100%**
Total % of nodes covered by ACO	**100%**	**100%**	**97.5%**
Total % of edges covered by ACO	**63%**	**33%**	**28%**
Total % of seeded faults covered by ACO	**97%**	**91%**	**82%**
Total test cases required to cover all the significant nodes, in case of ACO	**2**	**3**	**5**
Number of test cases required to cover all the nodes ,in case of ACO	**3**	**4**	**6**

T -3: Performance of ACO when nodes are deleted

5 Conclusion

The results of the proposed work will help tester's a lot during testing of the websites .With the help of automated and real test data acquired from the user sessions very less efforts are required by the testers with no manipulations. Various researchers quoted in their work that the test cases generated from user sessions are more practical, real and requires fewer efforts than that of other methodologies followed for testing the website in hand. With the help of proposed methodology these number of initial user sessions are significantly reduced to countable number to act as a testing data for test cases. With the execution of these numbers of test data, tester can become assured of at least those paths which have larger weights i.e, those paths which has been traversed by most of the users or the paths which contains nodes having very high degrees (significant nodes) or both. As said that 80 percent of the errors are found in 20 percent of the code, there is a very high probability that errors may exist in these paths and that can be easily unhide using the proposed approach. Maximum fault prone area of a website graph will be covered . There can be some of the paths (test cases) which will not be covered by generated test data, the probability of errors in these paths will be less, being less weighted, in case of time constraint, the tester can ignore these paths. If percentage of base request coverage during testing is one of the criteria of measuring efficacy of the work (=number of base requests in reduced test suite/number of requests in original test suite (which is nothing but all the user sessions retrieved from log file of the server)),the proposed methodology is achieving upto 100% base request coverage.

6 Limitations and Justifications

.One of the major limitations includes the websites under test are the projects of pre final year and final year students of bachelor of Computer Science and Engineering branch. These websites are not as large as that of real life websites like Alibaba, Amazon or Flipkart . Moreover the results of the work are not compared with any bench mark website. Results may vary due to approaches followed by professionally made websites and students made websites.

During the literature survey it has been found that scholars have talked about three types of dependencies named as link dependency, data dependency and functional dependency [13] during website/web application testing. In this work only link dependency and data dependency have been considered and no r eflections have been given to functional dependency.

Tracking user behaviour is one of the key issues in the proposed methodology. It has been found that researchers are now raising questions on quality of user sessions by just only considering the hits (page visited) .One of the major point which was missing, in user sessions based testing, was the time spend on each page, and this factor is assumed to be directly proportional to the interest of the user so this factor has been incorporated during the proposed work. But how much importance should be given to this factor is one of the issues which have remained untouched in the work. In this proposed work 25% weight age has been given. Another factor which can be incorporated in user behaviour is the total bandwidth consumed in particular session and on particular page also. Some tools have been found which will be useful to fetch this parameter from log file. This factor should also be given appropriate weight age. However this parameter is not considered during this work. Authors are proposing to work out on this parameter during future work.

All the faults are manually seeded and their representatives of real faults may be argued. All the faults have assumed to be of equal severity and equal execution time.

One may raise the question that everyone is talking about reduction of the test cases, prioritization of test cases, reduction of execution time of test suite

and removing the redundant and obsolete test cases then how this work is justified.

The justification is given as follows. Every independent path is a test case itself. There can be numerous number of test cases which depends upon the how much dense the website is. Each user session containing at least one branching node (page), is considered as execution of that particular test case (particular branch) with test data as name-value pair or connectivity. Many independent paths may not be covered in any of the user session.

Authors desire to rationalize that generating and executing each and every test case with every possible test data is next to impossible within given time frame and other constraints.

Authors want to validate that through the proposed approach not all the independent paths will be tested but the significant nodes and the corresponding path(s) will be tested.

Authors also wants to clarify that no doubt the execution time of each particular test case, generated by proposed approach, will be larger when comparing the original user sessions (which were of smaller size in terms of pages) and executing these user sessions by any replay tool. But if total time of executing all the original user sessions will be compared with reduced number of test cases, generated by the proposed approach, it will be significantly smaller and the objectives of the work will also be fulfilled.

Similar opinion will be given for redundant testing of the sub paths (part of independent path) within the website graph which are tested more than once. Because one of the objective was to at least test all the significant nodes present in the graph, the proposed work is successful in fulfilling that objective, without considering the redundancy.

References

[1] Elbaum, Karre and Rothermel (2003)Improving web application testing with user session data in Proceedings of the 25th International Conference on Software Engineering. IEEE Computer Society, pp. 49–59.

[2] Sampath, Sprenkle, Gibson, Pollock, and Greenwald (2007)Applying concept analysis to user-session-based testing of web applications, Software Engineering, IEEE Transactions on, vol. 33, no. 10,pp. 643–658.

[3] Sampath and Bryce (2012)Improving the effectiveness of test suite reduction for user-session-based testing of web applications Information and Software Technology 54 724–738 Elsevier

[4] Software Coverage :A Testing Approach Using Ant Colony Optimization By Sharma, Girdhar, Taneja, Basia,Vadla and Srivastava Springer-Verlag Heidelberg2011

[5]Optimized Test Sequence generation from Usage models using Ant Colony Optimization by Srivastava,Jose,Barade,Ghosh International Journal of Software Engineering and Application 2010

[6]An approach of Optimal Path Generation using Ant Colony optimization by Srivastava,Baby and Raghurama TENCON-2009

[7]Automated Software Testing using Metaheuristic Technique Based on An Ant Colony Optimization by Srivastava and Baby International Symposium on Electronic system design 2010.

[8]Analyzing Test Case Selection and Prioritization using ACO by Bharti Suri and Shweta Singhal ACM SIGSOFT November 2011.

[9] Test Case prioritization using Ant Colony optimization by Yogesh Singh,Arvinder Kaur and Bharti Suri ACM SIGSOFT July 2010.

[10] Structured Testing Using Ant Colony optimization by Praveen Ranjan Srivastava IITM December 2010 Allahabad.

[11] Bharti Suri and Shweta Singhal , " Understanding the effect of time-constraint novel technique for regression test selection and prioritization," International *Journal of System assurance Engineering and Managements*, Springer(2014). DOI: 10.1007/s13198-014-0244-3

[12] Maung and Win (2015)An Efficient Test Cases Reduction Approach in User Session Based Testing by International Journal of Information and Education Technology 2015.

[13]Garg,Dutta and ,French(2012),"New Test Case prioritization strategies for regression testing of web applications"International Journal of System Assurance Engineering and Management ,Springer.