I.J. Intelligent Systems and Applications, **2018, 2, 55-71**
Published Online February 2018 in MECS (http://www.mecs-press.org/)
DOI: 10.5815/ijisa.2018.02.06

A Novel Approach for Regression Testing of Web Applications

Munish Khanna
Department of Computer Engineering, YMCA University, Faridabad, India
E-mail: munishkhanna.official@rocketmail.com

Prof.(Dr.) Naresh Chauhan
Department of Computer Engineering, YMCA University, Faridabad, India
E-mail: nareshchauhan19@gmail.com

Prof.(Dr.) Dilip Sharma
Department of Computer Engineering, GLA University, Mathura, India
E-mail: Dilip.sharma@gla.ac.in

AbhishekToofani
Hindustan College of Science & Technology, Mathura, India
E-mail: abhishektoofany@gmail.com

Received: 23 March 2017; Accepted: 06 July 2017; Published: 08 February 2018

Abstract—Software testing is one of the most arduous and challenging phase which is to be implemented with the intention of finding faults with the execution of minimum number of test cases to increase the overall quality of the product at the time of delivery or during maintenance phase. With the ever increasing demand of web applications and to meet never ending customer expectations, updations are to incorporate which will be validated through testing process. The structure of the web applications (dynamic website) can be modeled using weighted directed graph which consists of numerous paths starting from homepage (index page) of the website. For thorough testing of the website each and every path of the graph should be tested but due to various constraints like time, money and human resources it becomes very much impractical. This scenario ultimately gives rise to the motivation for the development of technique which reduces the number of paths to be tested so that tester community can test only these numbers of path instead of all possible paths so that satisfactory number of faults can be exposed.

In this proposed approach assignment of weights on the edges of the directed graph takes place on the basis of the organization of the website, changes in the structure of the website at page level, experience of the coder and the behaviour of the users who have visited the website earlier. The most fault prone paths are identified using random, greedy, Ant Colony Optimization (ACO) and Artificial Bee Colony Optimization (ABCO) algorithms. Two small size websites and one company's website, and their two versions, were considered for experimentation. Results obtained through ACO and ABCO are promising in nature. This approach will support testing process to be completed in time and delivery of the updated version within given hard deadlines.

Index Terms—Artificial Bee Colony Algorithm, Ant Colony Algorithm, User Session based web testing, Web Application Testing, Test case Reduction and Regression Testing.

I. Introduction

In today's E-Commerce oriented state of affairs, web technology and its applications are playing very vast role. E-Commerce is mostly governed by the web applications which are based on different web technologies and hosted over the web servers so that user can access them over intranet or internet through web browser.

The structure of the web applications is complex and changeable in nature. Due to recurrent updates web application testing is required without interrupting the provided services. To achieve endless user expectations and sustainability in highly competitive environment, there is always a need of modernized, reliable and quality web application. Some published literature revealed that non availability of the renowned E-Commerce websites even for a short span of time results in huge losses for the company. Web application testing is executed with the objective of finding fault(s) at various levels (page, module or functionality) of the web application. Various web application testing strategies have been evolved but testing all of the web pages with every possible request (test data) i.e. thorough testing without interrupting the services is an exigent assignment due to constraints like monetary issues, short span of time and availability of skilled human resources.

The structure of the dynamic website resembles directed graph in which nodes of the graph represents the pages of the website while the edges of the graph represents link or data dependency between the pages. Every page is connected to at least one page thus it resembles connected graph. The structure of the website is complex in nature as it consists of huge number of paths generating from the home page. Traversing a particular path leads to the execution of all test cases which are needed to test the path. Due to enormous paths that must be traversed, for thorough testing, testing all the paths is troublesome assignment. This gives rise to the inspiration for the development of technique which reduces the number of paths to be tested by the tester. The criteria for selection of these paths is that they satisfy the characteristics of maximum possibility of existence of faults due to the organization of the website or changes made in the website and/or moreover these paths have high importance due to the broad interest of the users. Any fault on these paths will raise a question mark on the quality assurance of the software hence these paths have also named as fault prone paths. Testing only these paths will give assurance to the testers that although thorough testing is not performed but that part of the website is tested where there is maximum possibility of errors and/or where user's visited the most.

Inputs to the proposed framework are the structure of the website and the user's navigation behavior. Degree (in degree and out degree) of each page, addition of the new pages, deletion of the old pages, modification in the existing pages and the distance of the page from the root(home page of the website) are measured while considering the structure of the website. Modification is measured in terms of number of changed lines of code in the page.

Solely considering user session while testing has some relevant issues which need to be highlighted. Now a day's E-Commerce websites like flipkart, amazon and alibaba sells billions of products which give rise to huge number of pages in their websites. Most of the user's access very less percentage of the total website pages and situation may arise that some of the pages are never visited by any of the user and therefore it will not be traced from any user session fetched from logs of the website so these pages will never be tested. On the other side these pages may be modified and must have to be tested due to structural behaviour of the website and/or changes made in these pages. Similarly, considering only the structure of the website in mind, during testing, is not the best step. Testing that part of the website which is least or never visited by the users, in time constrained environment, is of less usage. Hence equal importance should be given to structure as well as behaviour for implementing hybrid testing, which is implemented in the proposed approach.

Behavior of the user is recorded in the form of user sessions and these recorded user sessions are considered as an execution of a particular path/sub path. Along with this, time spent on each page, activities performed on each page and bandwidth transferred on each page is also considered and calculated, while considering user behaviour. Thus user behaviour along with the structural property of the website acts as input to the system to build weighted directed graph thereafter four algorithms Random, Greedy, ACO and ABCO are applied to find set of minimal number of maximal fault prone paths from these inputs to meet out certain predefined objectives. To the best of author's knowledge very less work has been published in this area using these 04 algorithms, especially ACO and ABCO, for reducing the testing paths (and ultimately test cases) for web application testing.

The main contributions of the paper are as follows

- Calculating parameters related to the structure of the website and the user behaviour.
- Constructing weighted directed graph of the website, in hand, on the basis of calculated parameters.
- Finding the reduced number of test paths using four algorithms for two small size dynamic websites and one IT company's website using information computed in above discussed two points.

The organization of the rest of the paper is as follows: Section 2 describes published work related to problem, theory related to ACO and ABCO and published work using these two meta heuristic based search algorithms .Section 3 demonstrates all the setup required for experimentation in detail. Section 4 presents the experiments conducted, results, discussions and findings, if any, in detail. Section 5 concludes the work along with justification and limitations.

II. Related Works

This section is divided into three subsections where first two subsections present the application of ACO and ABCO in software testing while last part of this section depicts published literature that resembles problem in hand.

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 and quadratic assignment problem , distributed networks and data mining..

Sharma et al. [1] 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 et al.[2] 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, at least once.

Srivastava et al.[3] 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.

Srivastava et al.[4] 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.

Suri et al.[5] seeks the usage of ACO in reordering of the test suite in time constraint environment and analyze the behavior on eight programs under test.

Singh et al.[6] 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.[7] extends and improved the proposed approach of [2] 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. Bharti et al[8] presents the improved version of ACO for solving time constraint test suite selection and prioritization problem using fault exposing potential.

Yang et al.[9] proposed a comprehensive improved ant colony optimization(ACIACO) whose performance was compared on the basis of efficiency and coverage with genetic algorithm and random algorithm. The results of the proposed approach signify that the ACIACO can improve the search efficiency, restrain precocity, promote case coverage, and reduce the number of iterations.

The ABC was first proposed by in 2005 by Dervis Karaboga of Turkey and since from inception it has been widely accepted by the community of researchers and academicians. Academicians have started applying ABC on real life problems for achieving the better solutions with the help of it. Karaboga et al [10] presented a comprehensive survey of the Applications of ABC in a choice of engineering problems of various fields like Industrial Engineering, Mechanical Engineering, Electrical Engineering, Electronics Engineering, Control Engineering, Civil Engineering, Software Engineering Image Processing, data mining, Sensor networks, Protein structure and many more.

Various versions of the ABC have been proposed by the researchers which include Continuous ABC, Combinatorial/Discrete ABC, Hybrid ABC, Chaotic ABC, Binary ABC, Parallel and cooperative ABC and Multi-objective ABC.

Karaboga et al. [11] proposed the Combinatorial ABC for solving Travelling salesman problem which falls under the category of NP-Hard combinatorial optimization problem.

Lam et al.[12]compares the ABC with other Algorithms like Ant Colony Optimization(ACO) and Genetic Algorithm(GA). Authors proposed automatic generation of feasible independent paths with the help of ABC and then test suite optimization using the same algorithm.

Chong et al.[13] applied ABC algorithm for solving job shop scheduling problem which also falls under the category of NP-hard problems. The results of ABC were compared with ACO and tabu search to evaluate the performance of ABC. Kaur et al.[14] proposed ABC algorithm for regression test suite prioritization .The average percentage of conditions covered was the criteria to measure the efficacy of the proposed algorithm. The functioning of the algorithm was explained with the help of examples based on maximum code coverage.

Srikanth et al. [15] applied ABC to generate optimal number of test cases to be executed on the software i.e, generation of optimized test suite. Full path coverage has been assured by the approach.

Joseph et al [16] blend the Particle Swarm optimization with ABC and named as Particle Swarm Artificial Bee Colony algorithm (PSABC) for test case optimization. PSABC prioritizes test cases to reduce time and cost of regression testing. Maximum statement and fault coverage with in minimum execution time was the main objective of the proposed work.

Mala et al [17] proposed a framework for software test suite optimization using ABC approach. Anticipated Sequential ABC and parallel ABC were compared with random testing and GA on the coverage based test adequacy criteria. The performance of sequential ABC was poor then that of the GA while parallel ABC generates global or near-global optimal results for test suite optimization and it also converges within less iterations.

Dahiya et al[18]applied ABC for structural testing of ten real world programs. Results were not satisfactory in the programs having large input domains and many equality based path constraints.

Konsaard et al [19] make use of ABC algorithm to prioritize test suites based on code coverage. The revealed results were promising in nature and helpful in prioritizing test suites. They also conclude that ABC order takes slightly longer than Optimal Order and twice of the time used by GA order, its coverage is as good as GA and Optimal orders.

During study of various research papers published during last decade it has been found that researchers are practicing to use user session data to create test cases for web application testing. Elbaum et al [20] proposed five approaches for test case generation and functional testing of web application using user sessions. He illustrates that the fault detection capability of test cases generated using user session data is almost equivalent to white box testing of same web application.

Sampath et al. [21] cluster user sessions using concept analysis and proposed selection of subset of user sessions based on a concept lattice by applying three heuristic approaches. One user session can be randomly selected from each cluster to become a test case. The main objective of the work was reduction of the user sessions. The eminence of the user sessions collected is directly proportional to the effective user session based testing. Hence the reduced set has the capability for detection if the original one also has.

Sampath et al.[22] prioritize the reduced test suite to increase its rate of fault detection. The ordered reduced test suite can be executed in time constrained situations, where, even if all of the test cases execution is not completed, the best test cases from the reduced suite will be executed. They proposed several heuristics to order reduced test suites using experimentally verified prioritization criteria in the domain of web applications. The test suites were assembled from refined user sessions and applied on to the application to find out discrepancy between expected and actual results.

Peng et al. [23] proposed an approach in which test cases are generated automatically using user session data and request dependency graph of the web application. Genetic algorithm is used to generate realistic test suite, by mixing different user sessions, to cover fault susceptible transition relations. They depict that with small size of test suite, proposed approach achieve higher path coverage, request coverage and fault detection rate than that of conventional user session based testing.

Qian et al.[24] proposed an approach for generation and optimization of test cases for web applications based on user sessions using genetic algorithm. Redundant user sessions were removed by the reduction process. To bring concurrency in testing, user sessions are divided into groups, on the basis of threshold value, and then prioritize the groups and test cases within the group. GA was used to optimize the results of grouping and optimization.

Elbaum et al..[25] proves that user session data can be used to produce the test suite for web application testing and the efficiency of the suite will be equivalent to white box testing. Usage of user session data in other relevant issues in testing web application was also discussed.

Liu et al [26] proposed a novel method for test case optimization which is based on user session clustering based on hierarchical clustering algorithm. Different clustered test suites were created and one representative test case is selected from each cluster as the representative of it for functionality testing of the web applications. The results obtained using proposed algorithm was encouraging while experimenting on traditional small size online book store website.

Muang et al.[27] also worked on test case reduction on the basis of entropy using user sessions retrieved from log files. URLs coverage, Reduction time and the Test case reduction rates were the parameters for finding the efficiency of the proposed algorithm. Proposed algorithm was experimented on two applications, one website and one digital library system, and it were proved that the 90% of the original test suite gets reduced.

Li et al. [28] et al. proposed an algorithm which uses user session data for generating test cases. Less than 3% of the user sessions were selected as test data by the proposed algorithm. K-meteroid algorithm was applied to cluster the test cases. It was proved that as the number of cluster increases the more code will be covered which ultimately increases fault exposing capability.

Sprenkle et al. [29] proposed an approach “concept” which analyze the user sessions and convert them into test cases. The proposed approach cluster user sessions that represents alike use cases. After that heuristic is applied for the selection of user sessions such that reduced test suite explore all the unique URL’s of the original test suite. Three requirement based reduction techniques were compared with three variants of the proposed technique on two web applications. The compared parameters of the study were time and space cost, fault detection effectiveness, program coverage and reduced test suite size.

III. Experimental Designs

This section is separated into five subsections. During first subsection all the information related to experimental setup is discussed. While in remaining four subsections relevant theory, if any, and the proposed algorithm of ACO, ABCO, Random and greedy approach are discussed.

A. Experimental Setup

Request dependency graph (RDG) of the website is a directed graph where web pages represent nodes and linking between the pages represents edges between the nodes. Path in a graph is represented using traversed nodes and the relevant edges. There can be multiple possible paths from a source node (index page) to any other node which the user visited last (logout or leave the website due to any reason). Traversing these paths leads to an execution of the test cases and multiple test data can be generated for the same. If weights can be assigned on the nodes of request dependency graph it will be converted into weighted directed graph where weight on the link defines the significance of link and connected nodes (nodes connected at both ends of that link). It is clearly stated that components of the system which have high execution probability or providing more services will inclined more towards failures and should be given priority while testing [49]. Therefore higher weighted links and their connected nodes must be evaluated during testing phase.

It is not possible to execute all possible test paths, so it becomes critical problem to find and execute reduced number of high weighted test path(s) (highly fault prone) in the weighted request dependency graph where repetition of nodes and edges can be allowed but repetition of cycle is not allowed. The union of these paths will cover portion of the graph (dependency graph) where the chances of fault occurrence is very high and that must be identified and tested at highest priority. The test cases which will cover these paths will be executed instead of whole test suite. Moreover, the proposed approach also tries to cover maximum number of high weighted nodes (also called as significant nodes or important nodes). Having very short span of time in hand, if tester executes only these countable numbers of test cases it can be supposed that major possible testing in constraint environment is accomplished.

Each user session (sample shown below) consists of the elements like user's ip address, session time and pages visited by users. These user sessions are refined to find out pages visited during these session along with interacted name-value pairs. The retrieved information will be used across many verticals. First, it shows the interest of the users in the particular page and the weights are assigned accordingly. 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 the higher will be the significance of it. Moreover this information is converted into weight of the node by adding it with other factors. Second, user session acts as initial solution for the proposed algorithms.

```
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)

Two dimensional matrix (user session vs page numbers) is used to store user sessions in the form of 1 and 0 in which 1 represents page pi visited during that session si otherwise 0.

The structure of the website resembles directed graph in which pages (or forms) represents the nodes of the graph and the data or link connectivity represents edges between the nodes. Every path which has to be tested starts from root node (home page) and ends at destination node. Any fault on a particular page, which is the part of the path, can interrupt the subsequent remaining path and ultimately one or more functionalities of the website. Distance of the node from the root (home page) ,dfr, is calculated using formula 1/(height from the root). During the calculation it is assumed that root is at height 1.Nodes at height 2 will have distance ½, nodes at height 3 will have distance 1/3 and so on. The fault on the root or nearby pages of the root will be propagated throughout the path and whole of the path may get severely affected. This effect will get start deteriorating as we move away from the root and becomes almost nullify when the end of the path is reached. Hence severity of the fault is considered to be inversely proportional to its distance from the root (home page)

Software Engineering emphasizes on concepts of coupling which plays a vital role among the modules of the software. Offutt et al.[30] in his study on presentation layers of web applications for testing stated that there exists three types of coupling among modules which are "tight coupling", "loose coupling" and "extremely loose coupling". Authors strongly emphasize on extremely loose coupling for the software like web applications. This study and the presented scenario give rise to the motivation for incorporating this parameter (extremely loose coupling) in the proposed model. In the proposed work coupling is defined, at page level, as interdependency among web pages of website. Many web pages can be part of single module and single web page can be part of various modules. It is calculated as sum of in degree and out degree of the page. In a coupled system like dynamic websites the fault on a particular page pi will affect the expected output of those pages which are calling pi, moreover it may also temper the results when faulty page pi calls other non faulty pages. Hence it may highly prone to fault and may affect called and calling pages both.

Tracking user behavior is one of the key evidence used in this proposed methodology. User behaviour analytics plays a key role in enterprise management, marketing, risk and traffic analysis. In the proposed work, we have use the aspect of data logging in which are 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).

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.

Other major parameters which are missing, in user sessions based testing, is the time spend on each page, bandwidth utilized, number of key board hits on the particular page and number of mouse clicks on a page. These parameters are directly proportional to the interest of the user and these factors are as important as page visited 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 all these parameters and motivates us to consider all these parameters during testing.

For the calculation of time spent on each page parameter, various readymade tools like Google Analytics, Stat Counter, Deep Log analyzer and Web log expert Lite version 8.6 were analyzed but none of them directly support requirement of the proposed work. Ultimately a server side script has been made and incorporated to calculate the time spend on each page for PHP based website. Moreover control panel of the website and Inspectlet tool is used for JSP based website. However there are various issues which were resolved while calculating time spent on each page. First issue is the time zone factor which means that users can visit the website from various countries having different time zones. This issue is resolved by the above discussed server side script and Inspectlet/control panel. Another issue is how much idleness of the user is allowed. That is if user is not making any mouse movement or not striking any key for a long period of time then what should be the interpretation that is whether the user is reading the content of the page or he/she is not available and leave the browser as such and does not navigate further. If user does not respond for a fix amount of time the session logs out and the corresponding time spent is recorded. There are various readymade third party tools available for the notifications of key and mouse movements like clicktale, crazyegg, mouseflow, mousestat, clickheat, clickmeter ,Inspectlet and many more. With the help of one of these tools the movement of the input devices has been noticed on the server side. For finding the bandwidth utilization weblog expert and web log explorer tool has been used. Normalized interaction summation on an ith page with both of these devices is represented as iwpi.

Another parameter considered while assigning weight to the node is the experience of the coder who have coded the page and/or incorporate changes in the page. This experience, ce , is again divided into two sub parameters, total experience ,te, and experience in this type of project, pe, . Coder having zero to five year experience have been assigned weight equals to ten, if experience lies between five to ten years in that case assigned weight will be five. In case of experience larger than this one will be assigned as weight. The clear intention is that larger the experience of the coder and better will be the quality of the code (lesser will be the chances of making error).ce is calculated as (te+pe)/2. If more than one coder is used to code/change the page then average of ce of each programmer has been considered. The relevant data was gathered from the development team of the software company.

Last parameter measured for assigning the weight to the node is the changes in the lines of code for the next version. Changes include addition, deletion or modifications made in the current code of the page, excluding comments. System utility tool can be used for this calculation. The parameter value is computed using the formula

CLOC= (Number of lines added + number of lines deleted +number of lines modified) / (Number of lines before alterations) *100

To calculate the weight of the link, average of adjacent node values are taken. Weight wij of the existing link between any two nodes i and j is calculated as

$$w_{ij} = \frac{v_i+v_j}{2} \tag{1}$$

Weight of each node is calculated as shown in equation 2 which assigns equal significance to structure, d_i, as well as user behaviour, e_i.

$$v_i = d_i + e_i \tag{2}$$

Here d_i defines the summation of four factors, on ith page, which are dependency (summation of in degree and out degree while considering data and link dependency) of the node dop_i, its distance from the root (home page) dfr_i, changes made in the page $CLOC_i$ and coder experience ce_i.

$$d_i = dop_i + dfr_i + CLOC_i + ce_i \tag{2a}$$

e_i defines user's behaviour which is the summation of four factors entropy of the node, time spent on the node(page) t_{si} , bandwidth spend on that page $band_{si}$ and interaction with peripherals iwp_i during ith page

$$e_i = -\Sigma p_i log_b p_i \ + t_{si} + band_{si} + iwp_i \tag{3}$$

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

The objective behind equation (2) is that, also discussed previously, equal importance should be given to structure and user navigation behavior.

Muang and Win et al. [31] in their empirical work also uses entropy for the reduction of test cases by applying user session based approach. Their approach reduces the number of sessions on the basis of entropy.

During study of the previous published literature it has been observed that most of the researchers have applied their proposed work/technique/methodology on the same standard website of online bookstore (OBS), consisting of very less number of pages, with available online source code. For the validation of the proposed work almost similar website of 10 pages and another one of 40 pages were considered .One professionally created website which is handing company internal management is also taken for evaluation of the proposed work. Second version of these three subject websites was released in which some of the pages were added, some were deleted and some modifications were made at code level among all the websites. Dependency graphs of 10 pages (Fig. 2), Company Information Tracking System (CITS) website (Fig.3) and 40 pages website after uupations (Fig. 4) are shown below. In Fig.4 purple colored small circles represent modified (insertion/deletion/updation) pages. Given below Fig.1 depicts the block diagram of our proposed work.

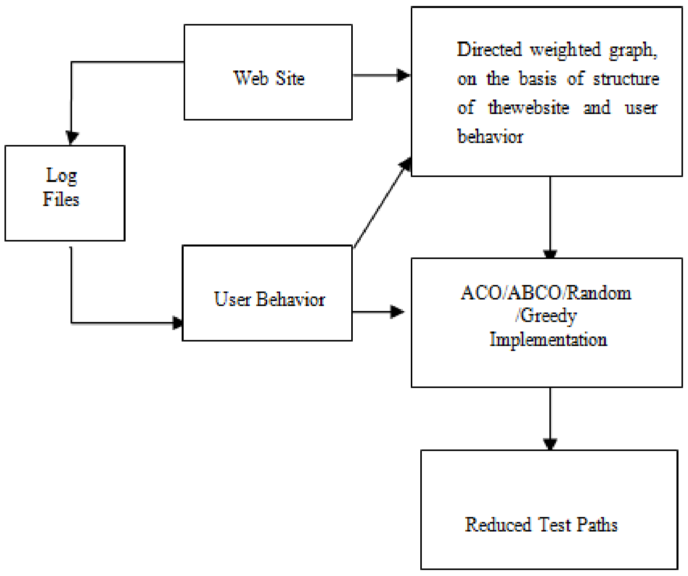
Web Site
Directed weighted graph, on the basis of structure of thewebsite and user behavior
Log Files
User Behavior
ACO/ABCO/Random /Greedy Implementation
Reduced Test Paths

Fig.1. Frame work of the Proposed Work

B. Applied Algorithms

In this remaining section all the four considered algorithms are discussed in detail. The common issue related to all these considered algorithms is when the algorithms are going to be stopped i.e. what should be the stopping criteria. There are various scenarios that satisfy the candidature properties to become stopping criteria. Out of these listed scenarios, which one encounters first, the algorithm will stop.

1. Either the dead end encounters.
2. Either all the pages are visited
3. Repetition of cycle begins.
4. Either all the significant nodes(pages) are visited.

In the proposed work one of the predefined objectives is traversing the paths having highest weights for which one of the followed approaches is the traversal of the significant nodes. In the weighted directed graph of the website, highest weighted 30%-40% of the total nodes are considered to be significant, depending upon the pages of the website. During testing phase these significant nodes must be evaluated, in spite of the location of these nodes. So it is expected that the output of the proposed approach,

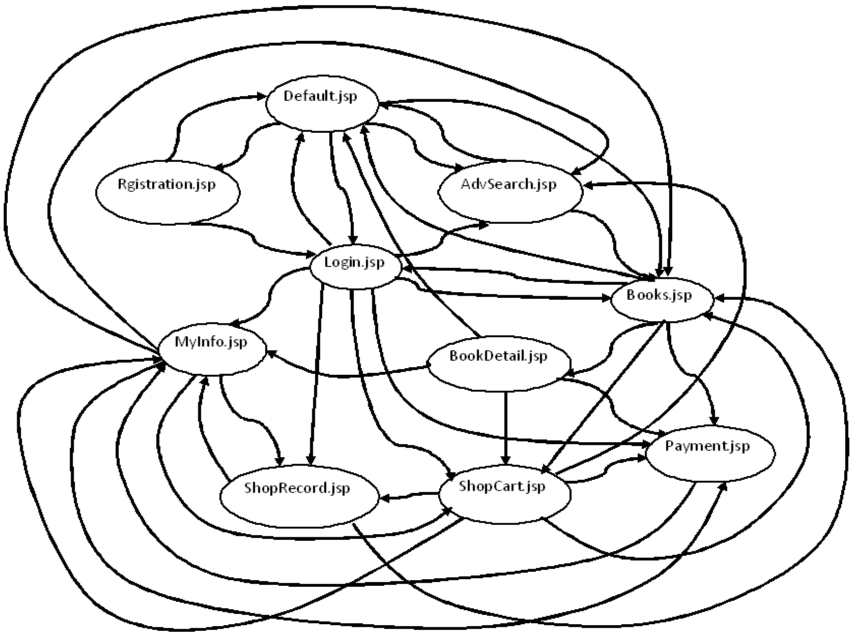

Fig.2. Dependency graph of 10 pages website

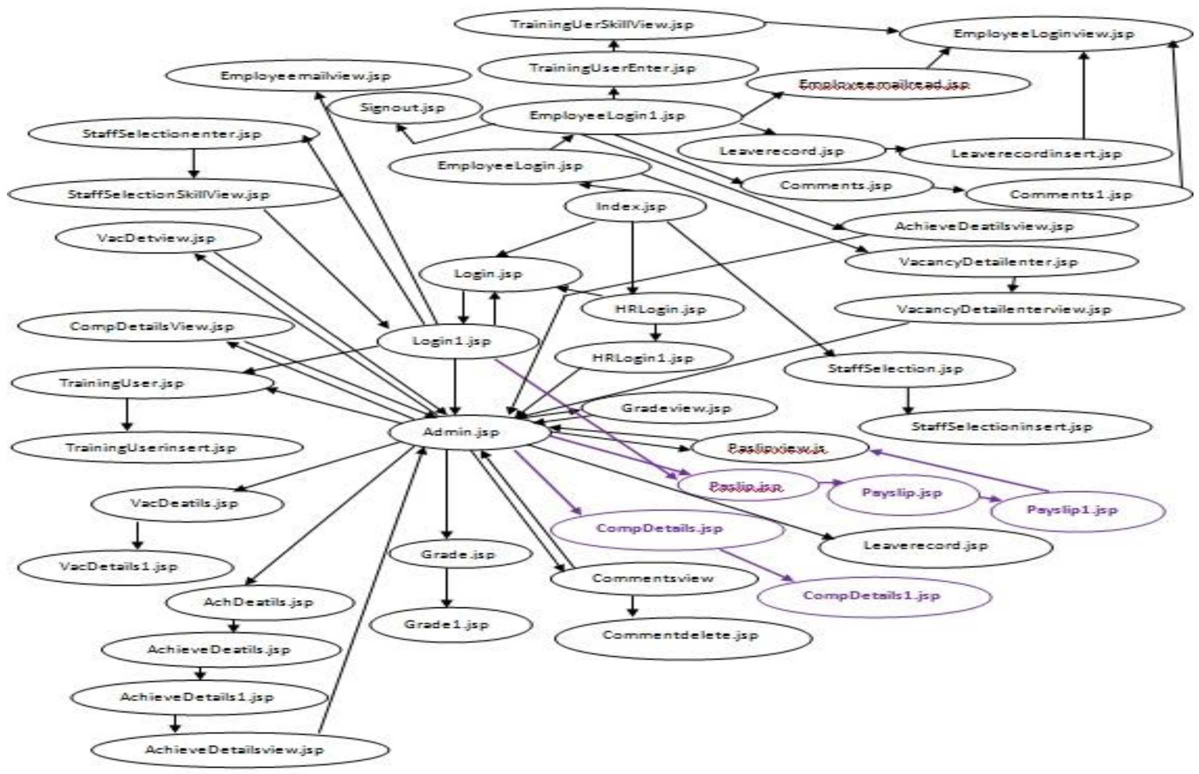

Fig.3. Dependency graph of CITS website

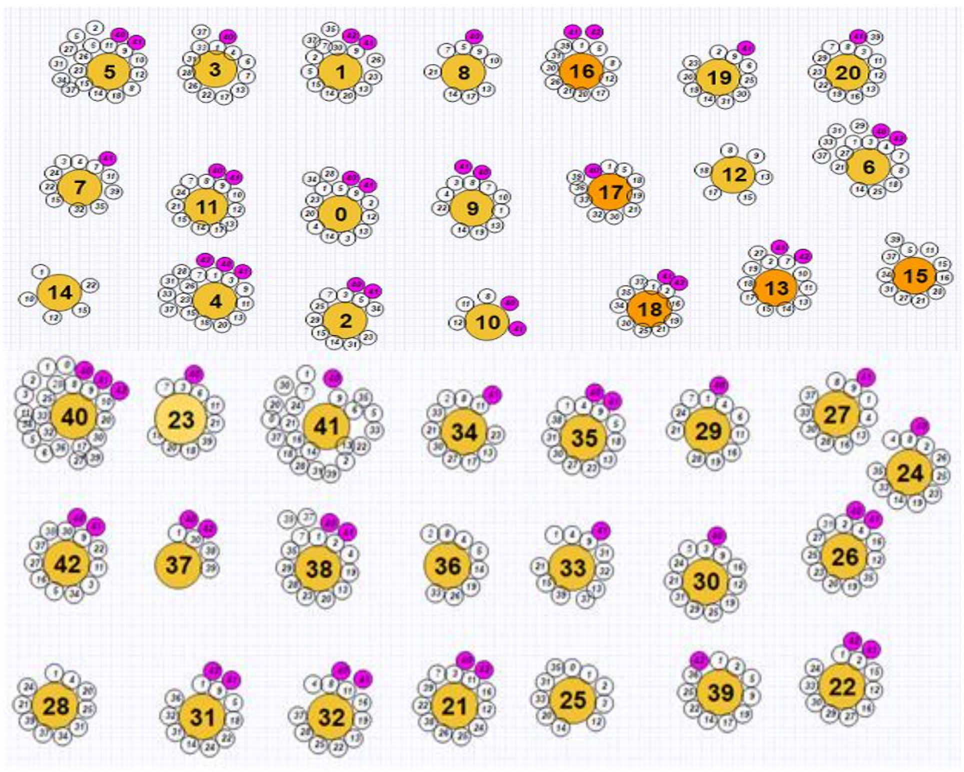

Fig.4. Graph of 40 pages PHP based modified website

which is identification and traversing of these high weighted test paths must satisfy this objective. It must also be shown in results that how many significant nodes are covered and how many test paths are required to cover these nodes.

B.1 Artifical Bee colony (ABC) Algorithm

ABC algorithm is inspired from the natural behavior of the bees and lies under the category of population based swarm intelligence approach. There are three types of bees (or agents) in the bee colony named as employer bee, onlooker bee and the scout bee. The roles of these types of bees are as follows .The employee bee acts as a search agent, the onlooker bee acts as a selector agent and scout bee plays the role of replacing agent. The general algorithm structure of the ABC optimization approach is given below

```
Initialization phase
Repeat
       Working of Employee Bees
       Working of Onlooker Bees
       Working of Scout Bees
       Memorize the best solution achieved
so far.
Till(Stopping criteria not reached)
```

B.1) Applying ABC in Web Application Testing

In order to reduce the number of test paths, identification of most fault prone paths and to find minimal number of fault prone test paths which substitutes thorough testing and enhance the confidence of the tester (by testing all significant nodes), ABC technique has been applied. Different phases of the proposed ABC are implemented as follows

Input to Algorithm:

- Two dimensional Adjacency matrix of size n by n (for :n" pages website) having value 1 if there exists link dependency or data dependency between pagea and pageb otherwise 0 .
- Two dimensional session matrix containing all the user sessions retrieved from the log file of the web server.
- 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 paths that would traverse through all significant nodes, almost all remaining nodes and ultimately high weighted paths.

Step 0: Preprocessing phase

The algorithm will be iterated twice the number of pages of the graph. Number of user sessions selected from the session matrix will be twice the number of pages in the website and these user sessions plays the role of initial food source for employee bees during first iteration. As already discussed, the user sessions vary in sizes which depends upon the number of pages accessed by the user during that session, so initial solutions vary in their sizes. As the user session consists of the web pages visited by the particular user during that session, so encoding technique used is discrete values in decimal numbers.

Some 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 with the help of roulette wheel, for the selection of next node, and the adjacency matrix of the website, for path verification. Remember the best solution found so far.

Step 1: Employee Bee Phase

The ultimate purpose of the Employee bee is the local search for a better solution in nearby areas. The solutions (initial sessions) are given to the employee bees. Now the employee bees start searching a neighbor source, named as X(k), of the particular session X(i) and gives rise to the new solution Y(i). Now evaluate the fitness of the original one and the new one Y(i) where X(i) ≠ X(k). Apply the greedy approach between these two. The new solution Y(i) from X(i) is calculated using equation (4)

$$Y(i) = X(i) + \emptyset * \big(X(k) - X(i)\big) \tag{4}$$

where $\emptyset$ is the random number between 0 and 1. For example suppose X(i)=0,2,32,12,11,12, now employee bee found the neighbor of the X(i) as X(k) where X(k)=0,2,32,12,11,12,22. Now set theory difference is applied to find out the difference between X(k) and X(i) which gives rise to 22 in this case .Now if there exists a path from 12 to 22 only then X(k) will be accepted otherwise find another neighbor X(k). Compare X(i)and Y(i). The solution having higher fitness will be chosen by the employer bee. Considering another example if X(k)= 0,2,32,12,11,12,22 ,27 and X(i)=0,2,32,12,17 , then the X(k) minus X(i) will be 22 and 27 , then we will start building the tour by

1. placing first 22 and then 27 after 17 in X(i) and verify if path exists using adjacency matrix
2. placing first 27 and then 22 after 17 in X(i) and verify if path exists using adjacency matrix
   (if both 1 and 2 options are possible then higher weighted path will be selected)
3. placing 22 after 17 in X(i) and verify if path exists using adjacency matrix
4. placing 27 after 17 in X(i) and verify if path exists using adjacency matrix
   (if both 3rd and 4th options are possible then higher weighted path will be selected)

Step 2: Onlooker Bee Phase

Calculate the probability of each of the solutions received from the employee bee by the formula

Probability p_i=weight of the solution i/weight of the best solution x(4a)

Then generate a random number (between 0 and 1) and compare the probability with this random number. If the solution (session) probability will be larger than that of the random number the session will be selected and stored. 40 solutions will be generated during this step. Then onlooker bee will again select the neighbor randomly from these selected solutions and try to generate a new solution Y(i), using equation (4)

Step 3: Scout Bee Phase

The threshold value is selected to discard the less profitable solutions. In this work the solutions which are having weight less than some threshold value are discarded and the solutions having weight larger than that of threshold will be selected by scout bee. Memorize the 5 best solutions found so far. Find the new solutions using roulette wheel equal to number of solutions discarded. Select best solutions among these and these will become food source for the employee bee and process move toward step number 1. This process will be repeated equal to number of pages in the website.

Best 05 results from each iteration will be stored in a file. From this file the best ones will be selected and will be the output of the whole ABC process.

B.1.1 Applying ABC in Regression testing of Web Application in deletion case

All the changes will be made in step 0 i.e, preprocessing phase of 4.1 while remaining all Employee bee phase, onlooker bee phase and scout bee phase remains as same as that of mentioned in 4.1. The rationalization of what extra has been done in the preprocessing phase is as follows. 03 nodes have been selected for the deletion numbered as 8, 22 and 34. Delete all the corresponding entries from the adjacency matrix and weight matrix with respect to these 03 nodes. 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 this case that 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 and added into pool of the food sources for employee bee.

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 and added into pool of the food sources for employee bee.

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.

B.1.2 Applying ABC in Regression testing of Web Application in addition case

Similarly in this case also all the changes will be made in step 0 i.e, preprocessing phase of 4.1 while remaining all Employee bee phase, onlooker bee phase and scout bee phase remains as same as that of mentioned in 4.1 . The explanation of what extra has been done in the preprocessing phase is explained with the help of scenario. 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. For this assign the weight equal to the highest weight of the node of the previous graph. Apply the roulette wheel for generating some initial solutions (food sources for employee bee) in which at least one of the newly added node 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 ABC algorithm which is "**Employee Bee phase**" .

B.2. Applying Ant Colony Optimization (ACO)

In ACO the behavior 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.

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

$$p_{ij} \leftarrow \frac{\tau_{ij}^{\alpha}.\eta_{ij}^{\beta}}{\Sigma\tau_{ij}^{\alpha}.\eta_{ij}^{\beta}} \qquad (5)$$

Here p denotes the probability to choose the path and τ denotes the pheromone value. Equation (5) 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-

$$\tau_{ij} \leftarrow (1-\rho).\tau_{ij} + \sum_{j=1}^{m} \Delta\tau_{ij}{}^{k} \qquad (6)$$

As said earlier ant selects high density pheromone path and updates the value of pheromone. So pheromone updation takes place as-

$$\tau_{ij} \leftarrow \tau_{ij} + \frac{Q}{l_i} \qquad (7)$$

Here Q denote constant and τ_i is pheromone value.η_{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}.

ACO is applied to reduce the number of test paths and are enough capable to meet out predefined objectives, discussed earlier. In this paper ACO is used to find the highly weighted paths (or tours) which start from source (index.jsp/home page). For the implementation of ACO fixed number of ants has been selected and the movement of these ants will be supported with some initial solutions in hand which are user sessions retrieved from session log file and that will guide the movement of the ants during initial phase.

In classical ACO technique 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} .

$$w_{ij} = \frac{1}{l_{ij}} \tag{8}$$

So eq-7 becomes:

$$\tau_{ij} \leftarrow \tau_{ij} + Qw_{ij} \tag{7a}$$

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

$$\tau_{ij} = 0.5 \quad \rho = 0.01 \text{ and } \alpha , \beta \, are \alpha = \beta = 1$$

Equation (7a) 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.

After encoding of problem, ACO is applied on the website under test, using following steps.

Input to Algorithm:

Same input as that of ABCO algorithm.

Output from the Algorithm

High weighted test paths traversing through significant nodes.

Step 1:

During this step initial pheromone, equals to 0.5, is laid down on the edges of the nodes of the website (graph) which are directly connected to each other. This step is executed before the ant's starts building their tour. During this phase different sessions retrieved from log file are analyzed and 3 best tours (user sessions) are selected 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. After first and subsequent iterations value of pheromone will get changed according to equations 6 and 7(a).

Step2:

Actual tour begins during this step. Number of iterations will be equals to 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(5) and returns next node to be selected for path generation. In this fashion paths of all the ants will be generated. It must be ensured that repetition of cycle is not allowed.

Step 4:

During this step, pheromone updation takes place as per the equations6 and 7(a). 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 all the nodes of the website (graph) including all significant nodes.

B.2.1 Applying ACO in Regression testing of Web Application in deletion case and addition case

The rationalization of what extra has been done when some pages are selected to be deleted is as follows. For example, in 40 pages graph 03 nodes have been selected for the deletion numbered as 8, 22 and 34. Delete all the corresponding entries from the adjacency matrix, weight matrix and pheromone matrix with respect to these 03 nodes and then apply all the six steps, mentioned above, of the algorithm.

Similarly if some pages are to be added to the existing website, in order to incorporate updated user requirements, update all the corresponding entries in the adjacency matrix. As new pages are not tested earlier hence there are very large chances that the fault(s) may occur in these added pages. Therefore they must be tested with highest priority and to implement highest priority, corresponding pheromone value is assigned with the highest available value in the pheromone matrix. On the similar lines, corresponding weight value is assigned with the highest available value in the weight matrix. After these updations, execute all the steps of the algorithm, mentioned above.

B.3. Random Approach and Greedy Approach

During implementation of random approach, the test path starts from the home page (or index page) till the stopping criterion is not achieved. To implement the randomness the following approach is followed, suppose if from page 'a', there exist four paths to reach different nodes, then random number is generated between one and four, next page will be visited on the basis of generated number and in this way the path will be traversed till the end.

In case of greedy approach next highest weighted available choice is considered till the stopping criterion is reached.

Here one important feature about both of these algorithms that should be mentioned here is that ABCO is implemented in such a way so that the longest paths should be identified while ACO works to find out most weighted paths.

IV. Results, Discussion and Comparison

Two websites, each consisting of 10, and 40 and pages respectively are created and tested to measure the performance of the proposed approach. One industry accepted professionally made jsp based website "Company Information Tracking System (CITS)" which consists of 65 pages and 44 modules (Fig.3) which is implemented in one of the company for its internal working is taken into consideration, whose figure is shown below. 02 of these websites (Fig.2 and Fig.4) were consists of jsp pages while one is based on PHP. Two websites were launched on the intranet of the college for thirty days. Staff members and students of pre final year and final year of various branches were requested to visit the websites so as to capture the behavior of different class of the users. Moreover one of the website is hosted on the internet (http://www.erphcst.top) for capturing user behaviour, fault seeding, making and hosting different versions and testing purpose. Faults mostly related to data dependency and link dependency, in equal ratio, was manually seeded, at random positions, into the running websites in the ratio of the number of pages. Larger the number of pages in the website, larger the fault seeded. Selenium testing tool is used for creating the test cases which can expose these manually injected faults hence only those faults were injected for which the test case can be created using this tool. The intension behind fault seeding is the validation of fault exposing capability of the algorithms and followed approach. After release of original version of every considered website, one updated version was released for each one. Many changes, discussed earlier, at page level were introduced .All the coding implementation of proposed algorithms is done in C#.

One of the thought process and perspective for solving this problem follows the path of optimization. The problem can be transformed to an optimization problem that can be solved using soft computing techniques .Suppose there are 40 pages in the dynamic website. In the simplest case, assuming every path begins at home page and finishes at particular page and not considering the repetitions of pages, total number of possible paths will be approximately 2^{38} (order of O(2^n)),where n is the total number of the pages),however some of the paths will not exist due to connectivity between the pages and some will overlap with each other also, hence testing each and every generated path will take approximately exponential complexity.

Published literature clearly indicates that meta heuristic techniques are capable of finding near optimal solutions of the problem having exponential complexity and the performance of meta heuristic is better than that of the other two. The same has been proved in the proposed work also.

For example(Table 1), suppose that there are 4 paths from node A of the graph to node F of the graph via nodes B, C, D and E and the weights assigned to the edges, are as follows.

The best possible, highest weighted, path which should be selected is A-E-F. This path will never be chosen by Greedy approach as it always selects A-B-F. Similarly in random approach the probability of selecting A-E is ¼ (not considering the impact of weight while selecting an edge). This scenario will occur many times while solving the instance of the problem hence finding optimal or near optimal using these two approaches is very less, while there are very high chances that the selected path using meta heuristic algorithm will be near optimal.

Table 1. Instance of weighted sub graph

Starting Node	Ending node	Assigned Weight
A	B	8
A	C	7
A	D	6
A	E	5
B	F	9
C	F	10
D	F	11
E	F	12

Table 2. Results of the experimentation process

Characteristics	Values for 10 pages Website	Values for 40 Pages Website	Values for CITS website
Total Number of User Sessions Considered	20	60	105
Total Number of Significant Nodes	4	16	33
Total number of nodes	10	40	65
Total number of edges	37	104	74
Total % of significant nodes covered by Greedy Approach	90%	87.5%	87.87%
Total % of significant nodes covered by ACO	100%	93.75%	93.93%
Total % of significant nodes covered by ABC	100%	87.5%	93.93%
Total % of significant nodes covered by Random Approach	80%	75.00%	72.27%
Total % of nodes covered by Greedy Approach	80%	75%	66.15%
Total % of nodes covered by ACO	100%	89.23%	84.61%
Total % of nodes covered by ABC	100%	90.00%	86.15%
Total % of nodes covered by Random Approach	70%	62.50%	53.84%
Total % of edges covered by Greedy Approach	81.08%	57.69%	60.81%
Total % of edges covered by ACO	89.18%	75%	81.08%
Total % of edges covered by ABC	91.89%	76.90%	82.43%
Total % of edges covered by Random Approach	70.00%	62.5%	64.86%
Total % of seeded faults exposed by Greedy Approach	80%	80%	85%
Total % of seeded faults exposed by ACO	100%	85%	88%
Total % of seeded faults exposed by ABC	100%	87%	89%
Total % of seeded faults exposed by Random Approach	80%	80%	84%
Total test cases required to cover >90% of significant nodes, in case of Greedy Approach	Less than 90% detected	Less than 90% detected	Less than 90% detected
Total test cases to cover >90% significant nodes, in case of ACO	1	5	6
Total test cases required to cover >90% significant nodes, in case of ABC	1	5	6
Total test cases required to cover >90% significant nodes , in case of Random Approach	3	8	15
Number of test cases required to cover >90% nodes ,in case of Greedy Approach	Less than 90% detected	Less than 90% detected	Less than 90% detected
Number of test cases required to cover >90% nodes, in case of ACO	2	7	9
Number of test cases required to cover >90% nodes ,in case of ABC	2	6-7	8-9
Number of test cases required to cover >90% nodes, in case of Random Approach	4	10	12
How many numbers of paths (test cases) generated, using Greedy Approach, lies in the top 10 weighted (manually verified) paths.	1	1	1
How many numbers of paths (test cases) generated, using ACO, lies in the top 10 weighted (manually verified) paths.	4	3	3
How many numbers of paths (test cases) generated, using ABC, lies in the top 10 weighted (manually verified) paths.	3	2	2
How many numbers of paths (test cases) generated, using Random approach, lies in the top 10 weighted (manually verified) paths.	3	1	1
How many numbers of paths (test cases) generated, using Greedy Approach, lies in the top 5 weighted (manually verified) paths.	2	1	1
How many numbers of paths (test cases) generated, using ACO, lies in the top 5 weighted paths.	3	2	2
How many numbers of paths (test cases) generated, using ABC, lies in the top 5 weighted paths.	2	2	1
How many numbers of paths (test cases) generated, using Random, lies in the top 5 weighted paths.	1	1	None

Above shown Table 2 clearly depicts the performance of various discussed algorithms on several parameters after running algorithms many times .Another issue which needs to be highlighted here is the source of test data generation for the test cases which will test these fault prone shortlisted paths. Actually the test case(s) may abstract in nature which will be converted into executable ones after adding corresponding values of parameters which has been retrieved from user sessions as name-value pairs during each request. In the published studies it has been concluded by the researchers that less human labors is required for the generation of test data and this generated test data will be more realistic and more effective to detects faults efficiently in nature as they are envoy of real user interaction. This will support the testers a lot and they do not require prior knowledge about structure of the code, underlying heterogeneous technologies and related hardware. These captured values will also support browser based inputs. However for websites which are larger in size and if some of the pages are not navigated during any session, then the corresponding test data can be generated manually.

In this last subsection the comparison between the proposed approach and relevant six published work has been made and shown in tabular format (Table 3) and explanation format.

One of the major findings of above comparison is that clustering is performed, in all the above papers except [25], in one way or other while finding the reduced number of test cases. Elbaum et al.[25] only talks about combining several user user's navigation path in different ways and proves that their testing approach as equivalent to white box testing. None of the above papers follows the approach of reduction of test cases using fault prone paths, recognized through ABCO and ACO, which are assigned using parameters discussed in this proposed approach.

All above referred papers consider only those URL's from the user sessions. However there may be the case, already discussed earlier also, that some of the URL's are not the part of any session, therefore remain isolated. However in the proposed work additional URL's are also considered.

In [25] it is clearly stated that user navigation behavior is not considered in their approach while in the proposed approach user navigation behavior is given 50% weight age while reducing the test suite. The above Table 3 clearly depicts that some of the considered parameters are not the part of their work. Above mentioned table and these discussed points clearly depicts that the approach and parameters considered in the above papers are majorly different from this work however the objective is same.

Numerous pages can be generated from the dynamic website depending upon the variety of inputs given at the interface which gives rises to pages explosion problem. In order to manage this problem, in this study, pages and forms as synonyms to each other while considering website's graph, where a form is a representative of page instances that may be generated from the same form (while considering diverse inputs from the users).

V. Conclusion, Limitation and Justification

A novel and quantified approach for web application testing is proposed in which three dynamic websites were selected as subject. Testing of the software is implemented while keeping one or more objectives in mind which are maximum code coverage/function coverage/block coverage/fault coverage/loop coverage. Path coverage is one of the criteria while testing. As the structure of the dynamic websites contains numerous paths, testing all the paths blindly is not an intelligent move. These paths are being assigned weights elegantly using various parameters to convert the graph into weighted one.

The proposed approach reduces the number of weighted paths smartly, using soft computing techniques, to be tested which is the proxy play to test case reduction, one of the strategies followed in regression testing [32,33,34,35,36,37]. This approach differs from other existing approaches in that it makes use of the structure of the website, changes made in the website and the behaviour of the end user to assign the weights to the paths and ultimately finding the most fault prone paths and mostly visited paths which need to be tested. Thus usage, frequency, traffic and internal structure all are taken into consideration while reducing the number of test paths. Some of considered parameters are still not applied in any of the published literature, as per best of author's knowledge. The approach also makes use of user sessions not only for recording user behaviour and calculating entropy but also for the initialization of ABCO algorithm and initial pheromone deposition and updation in ACO. Two versions of each website, under test, were considered for experimentation purpose.

The proposed algorithm tries to incorporate the features of both white and black box traditional testing approaches by giving equal importance to each of these. Testing of the software on the basis of structure and its related parameter correlates with white box testing and user's navigation behaviour and the associated parameters of the website correlates with black box testing.

The performance of four algorithms for solving the said problem were analyzed on various parameters As already stated that ACO was designed in such a way to explore the most weighted paths available in the weighted directed graph of website under test and the proposed wok was successful in achieving the objective. Similarly ABCO was able to discover the longest weighted paths. The performance of greedy and random approach was not able to beat the results achieved through ACO and ABCO.

The results of the proposed work will help tester's a lot during testing of the websites .Reduced number of test paths , automated and real data acquired from the user sessions using very less efforts makes the life of testers easier. Moreover the data being real as input by the users during surfing of the website no manipulations are required. In the constrained environment the support for maximum possible testing is performed using proposed approach. With the execution of these few paths, tester will be assured of at least those paths which are mostly navigated by the users and/or covering the complex parameters of the website thus the paths which should be given highest priority, during testing, are tested. There will be many paths which will not be traversed by the proposed approach, the errors may exist in these paths, but being less weighted they can be ignored due to limited time constraint to test all the paths. Thus the proposed work does not guarantee 100% fault coverage capability (one of the limitation). Moreover, faults are manually seeded in the websites considered for experimentation. One of the major limitations of the proposed work is that the websites under test are not as large as that of websites like Alibaba, Amazon or Flipkart .In this work link dependency and data dependency, at page level, have been considered but functionality dependency is not considered. One may raise the question that some portion of these shortlisted paths may overlap with each other and tested more than once. Authors want to justify that because testing all the significant nodes present in the graph and/or visiting all the pages, if possible, are the part of objectives of the proposed study, and may be tested more than once however the proposed work is successful in fulfilling that objective, without giving significance to some redundancy.

Table 3. Comparison of the proposed work with published literature

Reference	Parameters Considered				Publication	Remarks
	Data and Link Dependency (In Degree and Out Degree)	Distance from the root, coder experience and changes in LOC	User Sessions	Time Spend on each page , interaction with peripherals on the page and Bandwidth utilized during each page		
Elbaum et al.[25]	✓	X	✓	X	IEEE Transaction 2005	Adding/ merging of the user sessions
Sprenkle et al.[29]	✓	X	✓	X	IEEE Conference	Clustering approach
Liu et al.[26]	✓	X	✓	X	IEEE Conference 2011	Clustering approach
Li et al.[28]	✓	X	✓	X	Springer Verlag Heidelberg 2011.	Clustering approach
Maung et al.[27]	✓	X	✓	X	Springer Verlag Switzerland 2016	Clustering approach
Sampath et al.[22]	✓	X	✓	X	Elsevier Journal 2012	Heuristic approach
Proposed Approach	✓	✓	✓	✓		"Finding paths in the graph" based approach

REFERENCES

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

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

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

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

[5] Bharti Suri and ShwetaSinghal , "Analyzing Test Case Selection and Prioritization using ACO", ACM SIGSOFT November 2011.

[6] YogeshSingh,Arvinder Kaur and Bharti Suri , "Test Case prioritization using Ant Colony optimization", ACM SIGSOFT July 2010.

[7] Praveen Ranjan Srivastava "Structured Testing Using Ant Colony optimization",IITM December 2010 Allahabad.

[8] Bharti Suri, ShwetaSinghal, "Development and validation of an improved test selection and prioritization algorithm based on aco",International Journal of Relability,Quality and Safety Engineering Vol 1 No.6 World Scientific Publishing Company 2014.

[9] Shunkun Yang, Tianlong Man, and JiaqiXu, "Improved Ant Algorithms for Software Testing Cases Generation,"The Scientific World Journal, vol. 2014, Article ID 392309, 9 pages, 2014. doi:10.1155/2014/392309

[10] Derviskaraboga,Beyzagoremli, celalOzturk and NurhanKaraboga "A Comprehensive Survey :Artificial Bee Colony(ABC) and applications" ,Springer March 2012.

[11] DervisKaraboga and Beyzagoremli "A Combinatorial

Artificial Bee Colony Algorithm for travelling salesman Problem", INISTA IEEE2011.
[12] Lam,Raju,Kiran,Swaraj and Srivastava, "Automated Generations of Independent paths and test suite optimization using artificial bee colony", ICCTSD2011 Elsevier.
[13] Chong,Low,Sivakumar,Lay "A Bee Colony optimization for job shop scheduling" , IEEE2006.
[14] A Bee Colony Optimization Algorithm for code coverage test suite prioritization IJEST April 2011.
[15] Srikanth,Kulkarni,Naveen,Singh and Srivastava "Test Case optimization using Artificial Bee Colony Algorithm" , ACC 2011 Part III CCIS 192 Springer
[16] A hybrid Model of Particle Swarm optimization and Artificial Bee Colony Algorithm for Test Case Optimization,Elsevier.
[17] Mala,Mohan and kamalapriya, "Automated Software Test optimization framework-and artificial bee colony optimization-based approach" ,IET software 2010 Volume 4 Issue 5 pp 334-348
[18] Dahiya,Chhabra and Kumar "Application of Artificial Bee Colony Algorithm to software Testing" , 21st Australian Software Engineering Conference 2010 IEEE
[19] Konsaard and Ramingwong ,"Using Artificial Bee Colony for code coverage based Test Suite Prioritization", 2015 IEEE.
[20] Elbaum, Karre and Rothermel "Improving web application testing with user session data", Proceedings of the 25th International Conference on Software Engineering. IEEE Computer Society, 2003,pp. 49–59.
[21] Sampath, Sprenkle, Gibson, Pollock, and Greenwald "Applying concept analysis to user-session-based testing of web applications", Software Engineering, IEEE Transactions on, vol. 33, no. 10,pp. 643–658, 2007.
[22] Sampath and Bryce "Improving the effectiveness of test suite reduction for user-session-based testing of web applications", Information and Software Technology 54 (2012) 724–738 Elsevier
[23] Peng and Lu "User-session-based automatic test case generation using GA" International Journal of the Physical Science July 2011.
[24] Qian "User Session-Based Test Case Generation and Optimization Using Genetic Algorithm", Journal of Software Engineering and Applications 2010
[25] Elbaum S, Rothermal G Karre S and Fisher M "Leveraging User-Session Data to support Web application Testing" IEEE Transaction 2005
[26] Liu Y, Wang K, Wei W, Zhang B and Zhong H "User session based test cases optimization method based on Agglutinate Hierarchical Clustering" IEEE Conference 2011
[27] Maung Mon H and Win ThiK "Entropy based Test cases reduction algorithm for user session based testing", Advances in Intelligent systems and computing Springer Switzerland 2016.
[28] Li J and Xing D ,"User session data based web applications test with cluster analysis", CSIE 2011 Springer-Verlag Berlin Heidelberg 2011
[29] SprenkleS,Sampath S and Souter A "An empirical Comparison of test suite reduction techniques for user session based testing of web applications"
[30] Offutt, J. and Wu Y. "Modeling presentation layers of web applications for testing", Software System Model Springer, pp.257-280 2010.
[31] Maung and Win "An Efficient Test Cases Reduction Approach in User Session Based Testing", International Journal of Information and Education Technology 2015.
[32] Sudhir Kumar Mohapatra, SrinivasPrasad, "Finding Representative Test Case for Test Case Reduction in Regression Testing", IJISA, vol.7, no.11, pp.60-65, 2015. DOI: 10.5815/ijisa.2015.11.08
[33] ManikaTyagi, SonaMalhotra, "An Approach for TestCase Prioritization Based on Three Factors", IJITCS, vol.7, no.4, pp.79-86, 2015. DOI: 10.5815/ijitcs.2015.04.09
[34] Abhinandan H. Patil, NeenaGoveas, Krishnan, Rangarajan, "Regression Test Suite Prioritization using Residual Test Coverage Algorithm and Statistical Techniques", International Journal of Education and Management Engineering(IJEME), Vol.6, No.5, pp.32-39, 2016.DOI: 10.5815/ijeme.2016.05.04
[35] Neha Chaudhary, O.P. Sangwan, RichaArora, "Event-Coverage and Weight based Method for Test Suite Prioritization", IJITCS, vol.6, no.12, pp.61-66, 2014. DOI: 10.5815/ijitcs.2014.12.08
[36] Izzat Alsmadi, SaschaAlda, "Test Cases Reduction and Selection Optimization in Testing Web Services", IJIEEB, vol.4, no.5, pp.1-8, 2012.
[37] Samia Jafrin, Dip Nandi, Sharfuddin Mahmood, "Test Case Prioritization based on Fault Dependency", I.J. Modern Education and Computer Science, 2016, 4, 33-45 Published Online April 2016 in MECS (http://www.mecs-press.org/) DOI: 10.5815/ijmecs.2016.04.05.

Authors' Profiles

Munish Khanna is presently Assistant Professor, Department of Computer Science & Engineering, Hindustan College of Science & Technology, Mathura India. He is pursuing his Ph.D. from YMCA University of Science & Technology, Faridabad, India. He has a teaching experience of 14 years. He has conducted and participated in number of short-term courses, seminars, and conferences conducted at the national/international level. He is reviewer of two international journals. He has published papers in international conferences, and national and international journals of repute. He is member of professional technical societies of CSI and IEEE. His research interest includes algorithms, software testing, automata theory, applications of soft computing techniques in computer science.

Dr. Naresh Chauhan is presently Professor, Department of Computer Engineering, YMCA University of Science & Technology, Faridabad, India. He has a rich experience of 24 years in teaching and Industries. He has guided successfully 3 PhD scholars and guiding 5 Ph.D. Scholars on the topics of Software Testing, Agile Software Development and Internet & Web Technology. He has conducted, participated, and served as a resource person for a number of short-term courses, seminars, and conferences conducted at the national level. Dr. Chauhan has published 37 research papers in various national and international journals, and 52 research papers in various national and international conferences. He is also the author of two books. One is titled "Operating System Concepts", published from Oxford University Press, India, 2014 and the other is titled "Software Testing : Principles & Practices" published from Oxford University Press in 2010. His research interest includes Internet technologies, Software Engineering, Software Testing.

Dr Dilip Kumar Sharma is Senior Member of IEEE, IEEE-CS, IEEE-IAS, IEEE- YP and Senior Member of ACM, USA and also life member of CSI, IETE, ISTE, IE, ISCA, SSI. He has delivered/chaired more than 50 invited talks/guest lectures and chaired the technical sessions at various institutes/conferences. He has edited two books and worked as Guest Editor of International Journals of repute. He has organized more than 12 IEEE/CSI International/National Conferences and Workshops with the capacity of General Chair, Co-General Chair, Convener and co-convener etc. He has published more than 60 research papers in International Journals /Conferences of repute indexed in SCI, Scopus and DBLP databases and participated in 50 International/National conferences. Presently he is working as Professor in Department of Computer Engineering & Applications, GLA University, Mathura, U.P, India. He is Secretary IEEE Uttar Pradesh Section (Geographical boundaries are Uttar Pradesh, Utrakhand & Nepal), Vice Chairman IEEE U. P. Section SP/Computer Society Jt. Chapter. His research interests are Web Information Retrieval and Software Engineering. He is currently guiding 4 PhDs and numbers of MTech / Thesis, Seminars and BTech Projects.

Abhishek Toofani presently works as assistant professor in Hindustan College of Science and technology, Mathura. He received his B.Tech degree in Information Technology from Uttar Pradesh Technical University and M.Tech in Computer Science and Engineering from Dayalbagh Educational Institute, Agra. He has five years of experience in teaching. He has published many papers, in National and International journals and conferences. He is also member of IEEE. His area of interest are Soft computing, Digital Image Processing, Pattern Classification and Software Engineering.

How to cite this paper: Munish Khanna, Naresh Chauhan, Dilip Sharma, AbhishekToofani, "A Novel Approach for Regression Testing of Web Applications", International Journal of Intelligent Systems and Applications(IJISA), Vol.10, No.2, pp.55-71, 2018. DOI: 10.5815/ijisa.2018.02.06