International Journal of Digital Information and Wireless Communications (IJDIWC) 3(4): 451-459
The Society of Digital Information and Wireless Communications, 2013 (ISSN: 2225-658X)

An Effectiveness Test Case Prioritization Technique for Web Application Testing

Mojtaba Raeisi Nejad Dobuneh[1], Dayang N. A. Jawawi[2] and Mohammad V. Malakooti[3]
[1, 2] Department of Software Engineering, Universiti Teknologi Malaysia (UTM), Skudai 81310, Johor Bahru, Malaysia, [3] Department of Computer Engineering, Islamic Azad University, UAE Branch
rmojtaba2@live.utm.my[1], dayang@utm.my[2], malakooti@iau.ae[3]

ABSTRACT

Regression tests are executed when some changes are made in the existing application in order to check the negative impact of the changes in the rest of the system or on the expected behavior of other parts of the software. There are two primary options for test suites used during regression testing, first generate test suites that fulfill a certain criterion, or user session based test suites. User-sessions and cookies are unique features of web applications that are useful in regression testing. The main challenge is the effectiveness of average percentage fault detection rate and time constraint in the existing techniques. Test case prioritization techniques improve the performance of regression testing, and arrange test cases in order to obtain maximum available fault that is going to be detected in a shorter time. Thus, in this research the priority is given to test cases that are performed based on some criteria related to log file which collected from the database of server side. In this technique some fault will be seeded in subject application then applying the prioritization criteria on test cases to obtain the effectiveness of average percentage fault detection rate.

KEYWORDS

Software testing, web application, regression testing, prioritization test cases, clustering, user session

1 INTRODUCTION

Web applications have served as critical tools for different business. Failure of these critical tools means the loss of millions of dollars for the organizations that are using them [1], [2].
Web applications are presentations of organizations and in front of a large number of audience faces. Most of the web applications must run without any interruption during the day and night. This requires that Software testers to detect the software bugs and software engineers to fix those bugs immediately and release the new versions. Under such circumstances the execution of regression tests is performed in order to make the performance of the new version as per the requirements of the client or organization. The fixing of bugs in web applications requires a short time span, so the test suites can help the testers to perform its best in detection of new faults during the testing phase [3].
The purpose of regression testing is to provide confidence that the newly introduced changes do not obstruct the behavior of the existing, unchanged part of the software. Regression testing is one of the largest maintenance costs during the software development cycle. It is a complicated process for web applications based on modern architectures and technologies. User session based testing has the benefit that tests can be automatically constructed from web logs for use in regression testing and they contain sequences of actions that real users have performed. User session based test cases also have the benefit that testers do not need to specify input for test cases. For instance, web applications are accessible through the Internet and each http POST and GET request that a user makes is written to a log file. The logs can then be passed into test cases by using the IP addresses, cookies, and time stamp for each POST and GET request in order to identify the steps of each user and to create the respective test cases [4].

2 RESEARCH BACKGROUND

In simplest form we can easily execute all the existing test cases in the test suite without any extra handling. However, software engineers know about the gradually growing size of the test suites due to software modifications. Thus, executing the entire test suite would be very

expensive. This leads the software engineers to think about deploying efficient techniques to reduce the effort that is required for regression testing in different ways.

In the life cycle of an application, a new version of the application is created as a result of (a) bug fixes and (b) requirements modifications [5]. A large number of reusable test cases may be accumulated from different application versions suitable to test newer versions of the application. However, running all the test cases may take a significant amount of time. An example that may spend weeks in the execution of all the test cases of an earlier version [6], Regarding the time restrictions, software testers need the selection and ordering of a covering subset of test cases for execution.

The three major approaches for regression testing are test suite minimization, test case selection and test case prioritization [7]. Test case prioritization (TCP) helps us to find out the different optimal combinations of the test cases. A prioritization process is not associated with the selection process of test cases, and it is assumed that all test cases must be executed. But, it tries to get the best schedule running of test cases in a way that if the test process is interrupted or early halted at an arbitrary point, the best result is achieved in which more faults are detected. TCP is introduced by Wong et al. [8]. The general TCP criteria and techniques are described in the literature review [7]. Structural coverage is the most commonly used metric for prioritization [9].

The logic of this criterion is that faster structural coverage of the whole software code leads to maximum detection of faults in a limited time. Therefore, the aim of this approach is achieving higher fault detection rates, via faster structural coverage. Although the most common prioritization technique is about structural coverage in different forms, some prioritizations techniques are presented that have different criteria [10], [11], [12].

An approach which is presented by Sampath et al., for TCP in order to test the web applications with different user sessions of previous software versions is recorded. The best test cases for a web application are session-based due to the reflection of real user patterns, and make the testing process quite realistic [3]. User-Session based techniques are new light weight useful mechanisms of testing. Just applying for web applications, Automating test process is more feasible and simpler by user sessions.

In user-session approaches the collection of the interactions of the users with the server is collected and the test cases are generated using a suitable policy. Client's requests transported as URLs composed of page addresses and name value pairs are the data to be captured. These data that can be found in the log files stored in web servers or cookies left in clients machines. Captured data about user sessions can be used to generate a set of http requests and turn into a real test case.

The benefit of the approach is to generate the test cases without any awareness web application’s internal structure. The test cases that are generated by user sessions are not too much dependent on different technologies that are required for web based applications.

In Table 1 comprise the existing methods based on the user session base web application testing between the years 2001 till 2012.

Table 1. Test case prioritization techniques

Study	**(Rothermel et al., 2001) [6].**	**(Elbaum et al., 2002)[9].**	**(Srikanth et al., 2005)[11].**	**(Sampath et al., 2008)[3].**	**(Sampath and Bryce, 2012)[13].**
Description technique	The techniques used for test cases prioritize for regression testing, including criteria, such as coverage code	The technique is leverage captured user behavior to generate test cases	The technique is a Prioritization of Requirements for Test (PORT).	The technique applied several new prioritization criteria to test suites	The technique to be used both the reduction and prioritization for test case web application
Strength	Improved the rate of fault detection	Leads to a reduction in the amount of required tester intervention	Improves detection of severe faults during the regression testing	Increase the rate of fault detection	Increased the effectiveness of test suite reduction
Limitation	Not cost effective	Faults need to be seeded in the application.	Not effectiveness in time and cost	Considering the costs associated with the prioritization strategies	Using reduction test suite can be missing some part of the test case and consequently effect on the finding faults

3 PROBLEM STATEMENTS

The current researchers have been using reduction test suite or prioritization test suite for effectiveness of fault detection rate and time. This research will address the useful new technique offered by Sampath et al. named session based technique. Although in [3], both approaches were used to improve the effectiveness of user session based testing but the reduction test suites cause to omit some part of the test case and consequently effects on the fault detection in web application.

Thus, it is suggested to improve the method by proposing a new technique that uses clustering and prioritization together with applied criteria. In this research we have proposed a method that the effectiveness of clustering test suites can be further improved by ordering the clustered set of test cases as well as to show the criteria for the ordering [14]. Such an ordering would be beneficial to a tester who will be faced with limited time and resources but still can complete the testing process. Therefore the problem is to

verify the new technique for improving the effectiveness of fault detection rate and time of testing in session-based test case prioritization of web applications.

3.1 Research Questions

In this paper a new technique has applied into the all test cases which is collecting from log files of respect subject, book store portal. The research question is:

"How can we increase the effectiveness of current session based on test case prioritization techniques for the web applications?"

In order to answer the above question the following questions need to be answered:

Question 1: How can the new technique with prioritized test cases based on number of most common http requests in pages will improve the rate of average percentage fault detection (APFD)?

Question 2: What is the effectiveness of the new technique with an ordered length of http request chains to obtain a better APFD rate?

Question 3: How can we use the new technique with an order dependency of http requests helps to improve the rate of APFD?

3.2 Objectives

The permutation of test cases in a way that leads to faster detection of maximum available faults in a modified version of web application needs to find good criteria. The goal of this research is to propose a new technique which merges two approaches of prioritizing and the clustering test suite to improve one of test case prioritization techniques called session-based technique in web application regression testing.

The research aimed to improve the accuracy of existing test suites with respect to the effectiveness of time and the rate of fault detection. The following objectives need to be accomplished in order to achieve our goals:

1. To develop a new technique to prioritize the cluster test cases in the web application testing process.
2. To propose the applied criteria for the fault detection rate to improve the effectiveness of new technique.
3. To verify the technique for effectiveness of mixing the above strategies together.

3.3 Research Justification

User session used for web application testing and this approach has been accurate and adequate for dynamic web application domain, Therefore we conduct our research by using session based test case prioritization for web application testing. The lines of code for large web applications are in the millions and debugging and error detection for all these lines are time consuming. Thus, there will be so many object interactions that also required the interactions of users significantly. The automated testing becomes complicated due to the continuous maintenance process and due to the changes that occurred in the profiles of the users [15]. Most of the existing literature studies are the two approaches for test methods disjoint. While reduction techniques generate a smaller set of tests than the original suite, even though a reduced test suite can be so large that it cannot be executed completely under time constraints. In this paper we have proposed a method that the effectiveness of clustering test suites can be further improved by ordering the clustered set of test cases as well as to show the criteria for the ordering such an ordering would be beneficial to a tester who will be faced with limited time and resources but still can complete the testing process.

4 RESEARCH METHODOLOGIES

This section refers to the method that used to study the field of research, perform the job and explain the results. This research is conducted to improve the rate of fault detection in web applications. Based on proposed the new technique with three criteria for prioritization test cases, the first step is dedicated to collecting log

file from server side for subject Bookstore web application which choose as case study in this research. Online Book Store is a shopping portal for buying books. The online book store allows users to register, log in, browse for books, search for books by keyword, rate books, add books to a shopping cart, modify personal information, and log out. Book uses JSPs for its front end and a MySQL database for the back end.

The Bookstore application has introduced to under graduate students of University Teknologi Malaysia (UTM) for collect log files on server side. The log files have been collected for 60 days. Clustering the previous works and documents and gathering useful information for next steps is basically done in step 2. A new problem is defined and explained in step 3. This step includes the techniques and methods that used to solve the problem too. Getting to work and performing some real experiments and writing the results is the next step of our research shown in step 4. Finally, some comparisons are made between the results that we obtained and those previous works to reach the conclusion, which is explained in step 5. Thus, this project is planned to accomplish in 5 steps. The research procedure is based on the five main phases as shows in figure 1. In the first phase we have introduced literature review. The second phase is to find the problem by systematic mapping on web application testing and determine the problem statement in this domain. The third phase is conducted to propose solution then we check the rate of fault detection in phase fourth. Finally, the last phase is evaluated with case study that compares the average fault detection rate with the results of previous techniques.

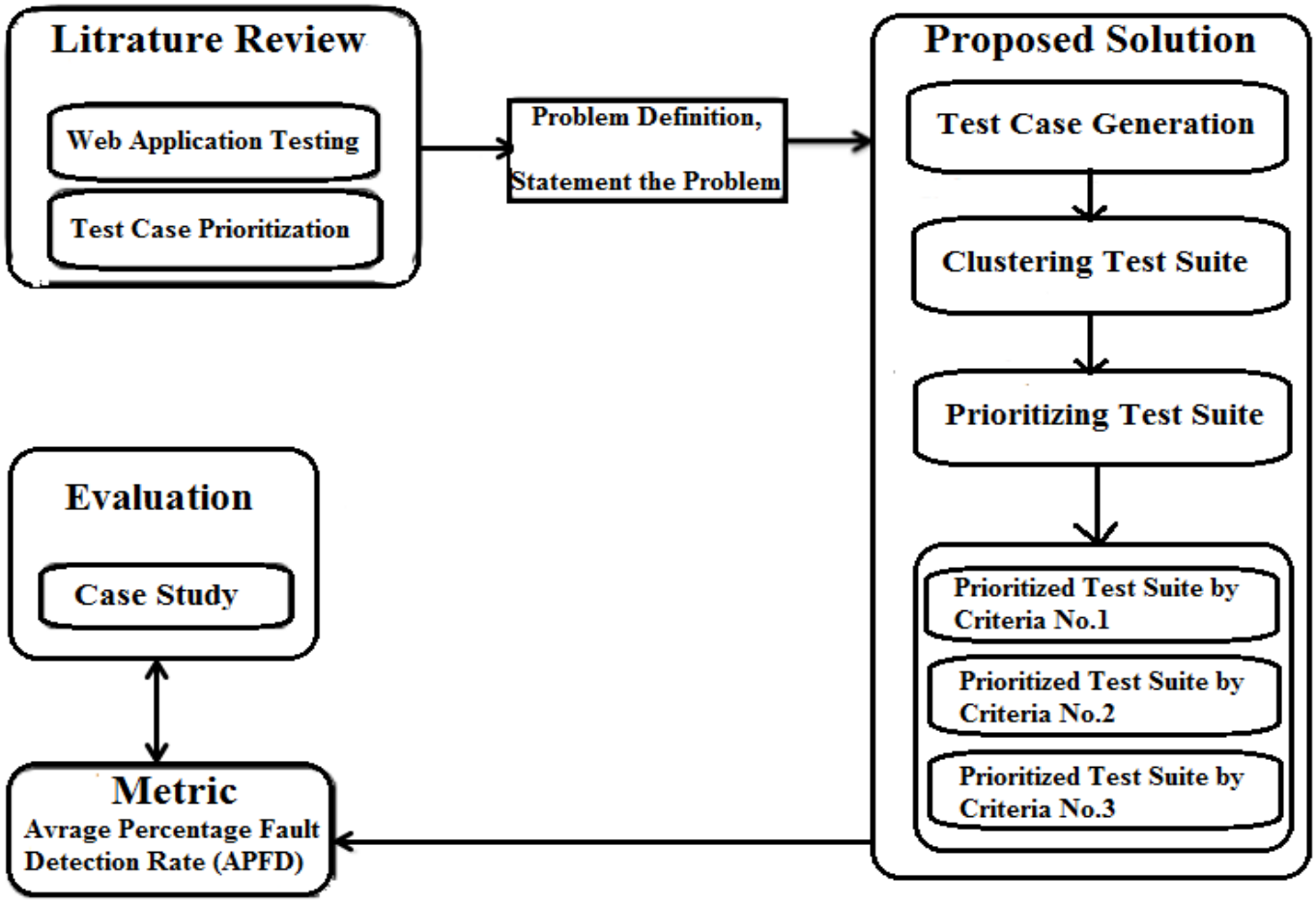

Figure 1. Research process

4.1 Literature Review

The focus of research is on two approaches in web application regression testing which are clustering test suite and prioritization test suite in case of effectiveness time and fault detection rate. This step is aimed to search about the current state of the art and challenges in web application test considering prioritization test case technique for better fault detection in less time. The drawbacks of these approaches are explored thoroughly in literature review step of the research process.

4.2 Problem Definition

The analysis of the problem, in test cases prioritization and web application testing is performed making the current literature review as a base for the issue or problem.

4.3 Proposed Solution

The goal of this research is to propose a new technique in test case prioritization for improving effectiveness of fault detection rate and time, due to these objectives the first step of the proposed solution is generating test cases base on the user session data which is using logs file in server side. Then we clustered the test suite into equivalence classes or groups of test suites in step 2. In the end we are prioritizing the test suite based on new criteria which are proposed in the research question.

For test case generation, the test cases come from real user session for web application testing, while using log file in sever side then convert each log file to the test cases.

Test Suite **c**lustering techniques could be categorized into modes partitioned or hierarchical. A partitional clustering algorithm constructs partitions of the data, where each cluster optimizes a clustering criterion, such as the minimization of the sum of squared distance from the mean within each cluster. The complexity of Partitional clustering is large because it enumerates all possible groupings and tries to find the global optimum. Even for a small number of objects, the number of partitions is huge. That's why; common solutions start with an initial, usually random, partition and proceed with its refinement. A better practice would be to run the partitional algorithm for different sets of initial points (considered as representatives) and investigate whether all solutions lead to the same final partition. Partitional Clustering algorithms try to locally improve a certain criterion. First, they compute the values of the similarity or distance, they order the results, and pick the one that optimizes the criterion [16]. Hence, the majority of them could be considered as greedy-like algorithms. Hierarchical algorithms create a hierarchical decomposition of the objects. They are agglomerative (bottom-up) or divisive (top-down):

(a) Divisive algorithms start with one group of all objects and successively split groups into smaller ones, until each object falls in one cluster, or as desired [17].

(b) Agglomerative algorithms follow the opposite strategy. They start with each object being a separate cluster itself, and successively merge groups according to a distance measure. The clustering may stop when all objects are in a single group or at any other point the user wants. These methods generally follow a greedy-like bottom-up merging.

Divisive approaches divide the data objects in disjoint groups at every step, and follow the same pattern until all objects fall into a separate cluster. This is similar to the approach followed by divide-and-conquer algorithms. For gathering all test cases into similarity groups have used K-mean clustering method.

The K-means algorithm comprises the following four steps:

1-Choose k initial cluster centers (representing the k transaction groups) randomly from the center of hypercube.

2- Assign all data points (representing the transactions) to the closest cluster (measuring from the cluster center). This is done by presenting a data point x and calculate the similarity (distance) d of this input to the weight w of each cluster center j. the closest cluster center to a data point x is the cluster center with minimum distance to the data point x.

$$d_j = \|x - w_j\| = \sqrt{\sum_{i=1}^{n}(x_i - w_{ij})} \quad (1)$$

3- Recalculate the center of each cluster as centroid of all data point in each cluster. The centroid c is calculated as follow:

$$\overrightarrow{C} = \langle W_1^C, W_2^C, \dots, W_n^C \rangle \quad (2)$$

Where

$$W_i^c = \frac{\sum_{j \in c} u_i^j}{N^c} \quad (3)$$

Where: N^C is the number of data points in the cluster

$$u_i^t = \begin{cases} 1, & if\ url_i \in t \\ 0, & otherwise \end{cases} \quad (4)$$

4-if the new centers are different from previous ones repeat step 2, 3 and 4 otherwise terminate the algorithm.

4.4 Prioritization

Test case prioritization (TCP) helps in finding out the different optimal combinations of the test cases. A prioritization process is not associated with the selection process of test cases, and it is assumed that all test cases must be executed, but it tries to best schedule running of test cases in a way that if the test process is interrupted or early halted at an arbitrary point, the best result that is finding more faults is achieved.

The three proposed prioritization criteria which are based on following: prioritization criterion with number of most common http requests in pages, prioritization criterion with an ordered length of http request chains, and prioritization criterion with an order dependency of http requests have been applied in this technique to obtain a better result.

4.5 Evaluation

The rate of fault detection is defined as the total number of faults detected in a given subset of the prioritized test case order [6]. For a test suite, T with n test cases, if F is a set of m faults detected by T, then let TF_i be the position of the first test case t in T', where T' is an ordering of T, that detects fault i. Then, the APFD metric for T' is given as:

$$\text{APFD} = 1 - \frac{\text{TF}_1+\text{TF}_2+\text{TF}_3+\cdots+\text{TF}_\text{n}}{\text{mn}} + \frac{1}{2\text{n}} \quad (5)$$

For 100% detection of faults, the time used by each prioritized suit is measured properly. The best possible option to calculate the detection of the total faults is to detect them in earlier stages of the tests.

5 Results

This phase of research methodology is comprised of verification and validation of the proposed new prioritization strategy for web based application. The subject book store is selected in order to verify it by applying criteria to test cases of web application.

The result of applied criteria to Bookstore shows in Table 2, as it can be seen the third criteria which ordered the test cases base on dependency http requests has been detected more fault at early stage of running test cases by 50 % of whole test cases detected about 94% of faults. The 93.33% percentage of total faults has been detected by first criteria (number of most common http requests in pages) with executed 60% of exist test cases. The length of http request chains prioritization criteria as a second criterion performed poorly for detected 94% fault by run more than 90 % of test cases.

Test cases executed by Random priorities create a reasonably effective test order with APFD comparable to the other techniques.

Table2: Average percentage of fault detection for Bookstore

Percentage of test cases execute	fault detected by first criterion	fault detected by second criterion	fault detected by third criterion	fault detected by random
10%	76.67	70	76.67	53.33
20%	83.33	80	76.67	53.33
30%	83.33	86.67	83.33	66.67
40%	83.33	86.67	83.33	80
50%	86.67	86.67	93.33	80
60%	93.33	86.67	93.33	80
70%	93.33	86.67	93.33	80
80%	93.33	93.33	93.33	86.67
90%	93.33	93.33	93.33	93.33
100%	93.33	93.33	93.33	93.33

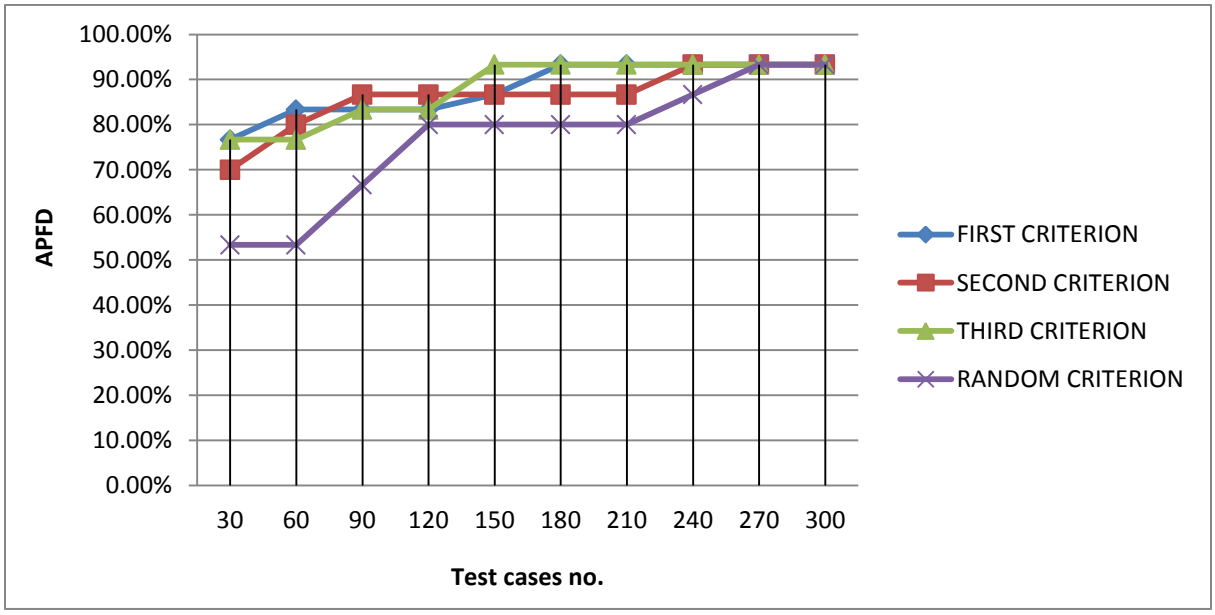

Figure 2. Results of APFD prioritization criteria

Figure 2 shows the graph of APFD by prioritization criteria in proposed technique to obtain the goals.

6 CONCLUSIONS

The web application domain has an advantage that actual user-sessions can be recorded and used for regression testing. While these tests are indicative of the user's interactions with the system, both clustering and prioritizing user-sessions has not been thoroughly studied. In this paper we examined a new technique for using cluster based test case prioritization of such user-sessions for web applications. By applying several new prioritization criteria to these test suites to identify whether they can be used to increase the rate of fault detection. Prioritization by frequency metrics and systematic coverage of parameter-value interactions has increased the rate of fault detection for web applications.

7 REFERENCES

1. Blumenstyk, M. "Web application development-Bridging the gap between QA and development", (2002).
2. Pertet, S. and Narasimhan, P, "Causes of Failure in Web Applications (CMU-PDL-05-109). Parallel Data Laboratory. 48, (2005).
3. Sampath, S., Bryce, R. C., Viswanath, G., Kandimalla, V. and Koru, A. G, "Prioritizing user-session-based test cases for web applications testing", Proceedings of the 2008 Software Testing, Verification, and Validation, 2008 1st International Conference on: IEEE, 141-150, (2008).
4. Sprenkle, S., Gibson, E., Sampath, S. and Pollock, L. (2005)." Automated replay and failure detection for web applications". Proceedings of the 2005 Proceedings of the 20th IEEE/ACM international Conference on Automated software engineering, ACM, 253-262.
5. Onoma, A. K., Tsai, W.-T., Poonawala, M. and Suganuma, H, "Regression testing in an industrial environment", Communications of the ACM. 41(5), 81-86, (1998).
6. Rothermel, G., Untch, R. H., Chu, C. and Harrold, M. J, "Prioritizing test cases for regression testing", Software Engineering, IEEE Transactions on. 27(10), 929-948, (2001).
7. Yoo, S. and Harman, M., "Regression testing minimization, selection and prioritization: a survey. Software Testing, Verification and Reliability", 22(2), 67-120, (2012).
8. Wong, W. E., Horgan, J. R., Mathur, A. P. and Pasquini, A, "Test set size minimization and fault detection effectiveness: A case study in a space application", Journal of Systems and Software. 48(2), 79-89, (1999).
9. Elbaum, S., Malishevsky, A. G. and Rothermel, G, "Test case prioritization: A family of empirical studies", Software Engineering, IEEE Transactions on. 28(2), 159-182, (2002).
10. Leon, D. and Podgurski, A," A comparison of coverage-based and distribution-based techniques for filtering and prioritizing test cases", Proceedings of the 2003 Software Reliability Engineering: IEEE, 442-453, (2003).
11. Srikanth, H., Williams, L. and Osborne, J, "System test case prioritization of new and regression test cases", Proceedings of the 2005 Empirical Software Engineering, International Symposium on: IEEE, 10 pp., (2005).
12. Tonella, P., Avesani, P. and Susi, A, "Using the case-based ranking methodology for test case prioritization", Proceedings of the 2006 Software Maintenance, 22nd IEEE International Conference on: IEEE, 123-133, (2006).
13. Sampath, S., Bryce, R, C., "Improving the effectiveness of test suite reduction for user-session-based testing of web applications", Information and Software Technology 54 (2012) 724–738.
14. Raeisi Nejad Dobuneh, M., Jawawi, N.A.D, Malakooti V.M, "Web application regression testing: A session based test case prioritization approach", The International Conference on Digital Information Processing, E-Business and Cloud Computing, 107-112, (2013).
15. Kirda, E., Jazayeri, M., Kerer, C. and Schranz, M, "Experiences in engineering flexible web services", Multimedia, IEEE. 8(1), 58-65, (2001).
16. Lazli, L., Mounir, B., Chebira, A., Madani, K. and Laskri, M.T., "Connectionist probability estimators in HMM using genetic clustering application for speech recognition and medical diagnosis", International Journal of Digital Information and Wireless Communications 1(1), 14-31, (2011).
17. Kaufman, L., Rousseeuw, P.J., "Finding groups in data: An introduction to cluster analysis", John Wiley and sons, (1990).