International Refereed Journal of Engineering and Science (IRJES)
ISSN (Online) 2319-183X, (Print) 2319-1821
Volume 6, Issue 2 (February 2017), PP. 63-69

Application of Genetic Algorithm in Software Engineering: A Review

Reena[1], Pradeep Kumar Bhatia[1]
[1]*Department Of Computer Science & Engineering Guru Jambheshwar University Of Science & Technology, Hisar(Haryana)*

Abstract. The software engineering is comparatively new and regularly changing field. The big challenge of meeting strict project schedules with high quality software requires that the field of software engineering be automated to large extent and human resource intervention be minimized to optimum level. To achieve this goal the researcher have explored the potential of machine learning approaches as they are adaptable, have learning ability. In this paper, we take a look at how genetic algorithm (GA) can be used to build tool for software development and maintenance tasks.
Keywords: Genetic Algorithm, Software Testing, Component Repository.

I. INTRODUCTION

Modern software is becoming more expensive to build and maintain. Software development management and software quality goals are necessary, but not competent for the needs of today's marketplace. Shorter cycle time, completed with least resources is also in demand [2].The challenge of developing software system in a fast movingEvolutionary Algorithms scenario gives rise to anumber of demanding situation. First situation is identifying software components is a crucial task in software development. The second one is to minimize number of test cases develop for the testing purpose. To answer the challenge, a number of approach can be utilized one such approach is the evolutionary algorithm [1]. By using evolutionary algorithm software is developed, modified and maintained at specification level, and automatically produced high quality software in shorter period [3].This evolutionary approach will enable software engineering to become the discipline capturing and automating currently undocumented domain and design knowledge [4].

In order to realize its full potential, there are tools and methodologies needed for the various tasks inherent to the evolutionary algorithm. In this paper, we take a look at how genetic algorithm can be used to build tool for software development and maintenance task as genetic algorithm have robustness and Genetic Algorithms are commonly used to generate high-quality solutions to optimization and search problems by relying on bio-inspired operators such as mutation, crossover and selection [1]. In this paper, we survey the existing work on application of GA in software engineering and provide research directions for the future work in this area.

II. GENETIC ALGORITHM (GA) METHODOLOGY

Genetic algorithms (Goldberg, 1989) in particular became popular through the work of John Holland [5] in the early 1970s, and particularly his book Adaptation in Natural and Artificial Systems (1975). Genetic Algorithms (GAs) are adaptive heuristic search techniques based on the evolutionary ideas of natural and genetic selection [6]. It represents an intelligent exploitation of a random search within a defined search space to solve a problem. Genetic algorithms are based on the principles of the evolution via natural selection, employing a population of individuals that undergo selection in the presence of variation- inducing operators, such as mutation and recombination. GAs is best used when the search space is large, complex and poorly understood, when domain knowledge is scarce or expert knowledge is difficult to encode. GAs also useful when there is a need to narrow the search space and in case of failure of traditional search methods [5, 6].

Algorithm for a GA is as follows [6]

```
Initialize (population)
Evaluate (population)
While (stopping condition not satisfied) do
{
Selection (population)
Crossover (population)
Mutate (population)
Evaluate (population)
}
```

The algorithm will repeat until the population has evolved to form a solution to the problem, Or until a maximum number of iterations have taken place (suggesting that a solution is not Going to be found given the resources available. Figure 1 depicts the steps involved genetic algorithm.

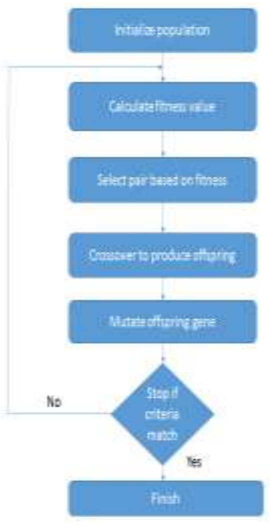

Figure1. Various Steps of Genetic Algorithm

1. Random population of n chromosomes is generated
2. Fitness value of each chromosome is evaluated
3. Create new population by applying genetic operators like Selection, Crossover, and Mutation etc.
4. New population generation is replaced.
5. If the specified condition is satisfied stop and return the solution.

III. SOFTWARE DEVELOPMENT LIFE CYCLE (SDLC) AND APPLICATIONS OF GAS IN SOFTWARE ENGINEERING

A variety of life cycle models has been proposed and is based on task involved in developing software [8]. Figure 2 shows SDLC/CBSD phases and applications of GAs in software engineering.

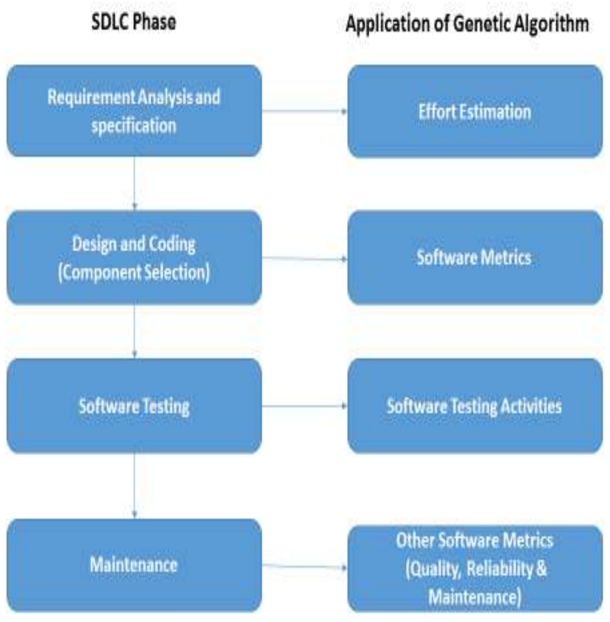
SDLC Phase
Application of Genetic Algorithm
Requirement Analysis and specification
Effort Estimation
Design and Coding (Component Selection)
Software Metrics
Software Testing
Software Testing Activities
Maintenance
Other Software Metrics (Quality, Reliability & Maintenance)

Figure 2. SDLC/CBSD Phase and Application of Genetic Algorithm

4 Applications of GAs in Software Engineering

Several areas in software development have already witnessed the use of GAs. In this section, we take look at some reported result of application of GAs in the field of software engineering. The list is definitely not a complete. It only serves as an indication that people realize the potential of GAs and begin to reap the benefits from applying them in software development.

4.1 Software Project Effort Estimation

Software cost estimation is one of the most challenging issues in software project development. To produce the accurate estimation, many models have been developed, but no model proves efficient with the uncertainty of the project development. Most of these models are based on the size measure, such as Lines of Code (LOC) and Function Point (FP) and Size estimation accuracy directly effect on cost estimation accuracy. As all we know the COCOMO model is the important model for Software Cost Estimation. Today's effort estimation models are based on soft computing techniques such as, genetic algorithm, fuzzy logic, neural network etc for finding the accurate predictive software development effort and time estimation.Genetic Algorithm can provide significant enhancement in accuracy and has the potential to be a valid additional tool for software effort estimation in large project. Genetic algorithm has been used for difficult numerical optimization problems and also used to solve system identification, signal processing and path searching problems [26].

Brajesh et al. proposed a model to estimate the software effort for projects sponsored by NASA using binary genetic algorithm. Modified version of the COCOMO model was provided to consider the effect of methodology in effort estimation. The performance of the developed model was tested on NASA software project data and the developed models were able to provide good estimation capabilities [27].

Vishaliet et al. proposed algorithm (GAs) was tested and the obtained results were compared with the ones obtained using the current COCOMO model coefficients. The results of the experiment show that in most cases the results obtained using the coefficients optimized by the proposed algorithm are close to the ones obtained using the current coefficients. Comparing organic and semi-detached COCOMO model modes, it can be stated that use of the coefficients optimized by the GA and ACO in the organic mode produces better results in comparison with the results obtained using the current COCOMO model coefficients [28].

Asthaet et al. proposed Genetic Algorithm (GAs) is tested on TURKISH and INDUSTRY dataset and the obtained results are compared with the ones obtained using the current COCOMO II PA model coefficients. The proposed model is able to provide better estimation capabilities. It is concluded that, By comparing the results, it can be stated that having the appropriate statistical data describing the software development projects, GAs based coefficients can be used to produces better results in comparison with the results obtained using the current COCOMO II PA model coefficients. The results also show that in most cases the results obtained using the coefficients optimized with the propose algorithm are close to the ones obtained using the current coefficients. The results also prove that in most cases the results obtained using the coefficients optimized with the propose algorithm are less than the real effort values [29]. Isa et al. have proposed a hybrid model based on GA and ACO for optimization of the effective factors' weight in NASA dataset software projects. The results show that the proposed model is more efficient than COCOMO model in software projects cost estimation and holds less Magnitude of Relative Error (MRE) in comparison to COCOMO model [30].

4.2 Software Metrics (Design and Coding)

Software metrics are numeric value related to software development. Metrics have traditionally been consisting through the definition of an equation, but this technique is limited by the fact that all the interrelationships among all the parameters be fully understood. The aim of research is to find the alternative methods for generating software metrics. Deriving a metrics using a GAs has several advantages [12].

R Vankudothet et al. work on selection of system software component. It is an important decision of design stage and has a significant impact on various system quality attributes. To determine system software component based on architectural style selection, the software functionalities have to be distributed among the components of software. The author present a method based on the Genetic Algorithm that use cases the concept and design procedure of Genetic Algorithm as techniques is proposed to identify software components and their responsibilities. To select a proper Genetic Algorithm method, first the proposed method is performed on a number of software systems using different Genetic Algorithm methods, and the results are verified by expert, and the best recommended. By sensitivity analysis, the effect of features on accuracy of Genetic Algorithm is evaluated then Finally determine the appropriate number of Genetic Algorithm (i.e. the number of software components), metrics of the interior cohesion of Genetic Algorithm and the coupling among them are used[31].
CBSD is used to reduce software development time by bringing the system to markets as early as possible. CBSD process consists of four major processes: component qualification, component adaptation, component composition and component update [10]. To realize the benefits which CBS brings it is imperative that the right software component is selected for a project, because selecting inappropriate component may results in

increased time and cost of software development but CBSD aims at reducing [11, 12]. Component selection is a major challenge to CBS developers, due to the multiplicity of similar components on the market with varying capabilities. Several approaches and criteria have been proposed for component selection, there is no well-defined procedure to select optimized components. K Vijayalakshmiet. al has given an automated approach based on Genetic Algorithm that enables the selection of software components both considering functional and non-functional requirements to find the best combination of components [9], [10], [11], [12].

Seyed Mohammed et al. propose a novel GA-based algorithm (Genetic Algorithm) as a powerful optimization search algorithm, called SCI-GA (Software Component Identification using Genetic Algorithm), to identify components from analysis models. The SCI-GA uses software cohesion, coupling, and complexity measurements to define its fitness function. For performance evaluation, the algorithm SCI-GA is evaluated using three real-world cases. The results show that SCI-GA can identify correct suboptimal software components, and performs far better than alternative heuristics like k-means and FCA-Based methods [1].

Kwonget.et al. has given the formulation of an optimization model of software components selection for CBSS development. This model has two objectives: maximizing the functional performance of the CBSS and maximizing the cohesion and minimizing the coupling of software modules. A genetic algorithm (GA) is used to solve the optimization model for determining the optimal selection of software components for CBSS development. It was prove by giving an example of developing a financial system for small- and medium-size enterprises is used to illustrate the proposed methodology [10].

Saxsena et al. an attempt to throw light which on the one of the major issue of component based software engineering is concerned with the “Component Selection”. Genetic Algorithms based approach is used for component selection to minimize the gap between components are selected [11].

4.3 Software Testing Activities

Software testing is the process of executing a program with the intention finding bugs. Software testing consumes major resource in term of effort, time in software product’s lifecycle. Test cases and test data generation is the key problem in software testing and as well as its automation improves the efficiency and effectiveness and lowers the high cost of software testing. Generation of test data using random, symbolic and dynamic approach is not enough to generate optimal amount of test data. Some other problems, like non-recognition of occurrences of infinite loops and inefficiency to generate test data for complex programs makes these techniques unsuitable for generating test data. That why there is need for generating test data using search based technique. In addition to these there is also need of generating test cases that concentrate on error prone areas of code [13], [14], [15], [16].

The application of Genetic Algorithm in Software Testing is a new area of research that brings about the cross fertilization of ideas across two domains. Genetic Algorithm is used to generate test cases while ensuring that the generated test cases are not redundant. It maximizes the test coverage for the generated test cases. In order to carry out the effectiveness of the test cases and test data the quantification, measurement and the perfect modeling is required which is done by using the accurate suite of software test metrics. The test metrics are used to measure the number, complexity, quality. Abhishek et.al applied the optimization study of the test case generation based on the Genetic Algorithm and generates test cases which are far more reliable [17], [18].

By examining the most critical paths first, obtain an effective way to approach testing which in turn helps to refine effort and cost estimation in the testing phase. The experiments conducted so far are based on relatively small examples and more research needs to be conducted with larger commercial examples.Yang et. al introduce an approach of generating test data for a specific single path based on genetic algorithms. The similarity between the target path and execution path with sub path overlapped is taken as the fitness value to evaluate the individuals of a population and drive GA to search the appropriate solutions. The authors conducted several experiments to examine the effectiveness of the designed fitness function, and evaluated the performance of the function with regards to its convergence ability and consumed time. Results prove that the function performs better as compared with the other two typical fitness functions for the specific paths employed by the authors [19], [20].

Aladeen et al[14] have compared the software test data for automatic path coverage using genetic algorithm with Yong [20] for generating test data of path testing. They found GAs is useful in reducing the time required for lengthy testing by generating the meaningful test cases for path testing. The GAs is required to be built for structural testing for reduce execution time by generating more suitable test cases.

Roy et al. propose a technique that uses a Gas for automatic test – data generation. A GAs is a heuristic that mimics the evolution of natural species in searching for the optimal solution to a problem. In the test-data generation application, the solution sought by the GAs is test data that causes execution of a given statement, branch, path or definition-use pair in the program under test. The test data generation technique was implemented in a tool called TGen in which parallel processing was used to improve the performance of the search. To experiment with TGen, a random test data generator called Random was also implemented. Both TGen and Random were used to experiment with the generation of test data for statement and branch coverage of six programs [41].

Rajappa et al.proposed graph theory based on genetic approach to generate test cases for software testing. In this approach the directed graph of all the intermediate states of the system for the expected behaviour is created and the base population of genetic algorithm is generated by creating a population of all the nodes of the graph. A pair of nodes referred to as parents are then selected from the population to perform crossover and mutation on them to obtain the optimum nodes. The process is continued until all the nodes are covered and this process is followed for the generation of test case in the real time system. The technique is more accurate in case of network testing or any other system testing where the predictive model based tests are not optimized to produce the output [15]. Parveenand Tai have demonstrated that it is possible to apply Genetic Algorithm techniques for finding the most critical paths for improving software testing efficiency. The Genetic Algorithms also outperforms the exhaustive search and local search techniques and in conclusion, by examining the most critical paths first, we obtain a more effective way to approach testing which in turn helps to refine effort and cost estimation in the testing phase [42].K Singh used Genetic algorithm in scheduling of tasks to be executed on a multiprocessor system. Genetic algorithms are well suited to multiprocessor scheduling problems. As the resources are increased available to the GAs, it is able to find better solutions in short time. GAs performs better as compared to other traditional techniques. So GAs appears to be the most flexible algorithm for problems using multiple processors. It also indicates that the GAs is able to adapt automatically to changes in the problem to be solved [24].

4.4 Other Software Metrics (Quality, Reliability and Maintenance)

Garvin describes quality from five different views: transcendental view, user view, manufacturers view, product view and value based view. Quality must be monitored from the early phases to final phase such as analysis, design, implementation and maintenance phases. There are many quality models given, some of the standard models are listed here: McCall's model (1979), FCMM model and Bohem's model. McCall's model contains 11 attributes, out of which two are described here such as reliability and maintenance [33].

M Amoui et al. work for Improving software quality. It is a major area in software development process. Despite all previous attempts to evolve software for quality improvement, these methods are neither scalable nor fully automatable so in this research authors approach software evolution problem by reformulating it as a search problem. For this purpose, author apply software transformations in a form of GOF patterns to UML design model and evaluated the quality of the transformed design according to Object-Oriented metrics, particularly 'Distance from the Main Sequence'. This research based formulation of the problem enables us to use Genetic Algorithm for optimizing the metrics and find the best sequence of transformations. The implementation results show that Genetic Algorithm is able to find the optimal solution efficiently, especially when different genetic operators, adapted to characteristics of transformations, are used [34].

D M Thakore et al. work on the security issues by using GAs. Assigning access specifier is not an easy task as it decides over all security of any software though there are many metrics tools available to measure the security at early stage. But assignment of access specifier is totally based on the human judgment and understanding .Objective of Secure Coupling Measurement Tool (SCMT) is to generate all possible solutions by applying Genetic Algorithm (GA). It is quietly different than any other security Measurement Tool because it filters input design before applying metrics by GA.SCMT uses coupling, also feature of OO design to determine the security at design level. It Takes input as a UML class diagram with basic constraints and generates alternate solutions. Tool also provides metrics at code level to compute the security at code level. The result of both the metrics gives proof of secure design [35]. S H Aljahdali use GAs as powerful technique to estimate the parameters of well known reliability model. Software reliability models are useful to estimate the probability of the software fail along the time. Several different models have been proposed to predict the software reliability growth (SRGM); but none of them has proven to perform well considering different project characteristics. The ability to predict the number of faults in the software during development and testing phases.GAs is a powerful machine learning technique and optimization techniques to estimate the parameters of well-known reliably growth models. Moreover, machine learning algorithms, proposed the solution to overcome the uncertainties in the modeling by combining multiple models aiming at a more accurate prediction at the expense of increased uncertainty [36]. Baqais et al.[38] used GAs for estimating maintenance effort and cost. Maintenance is an important activity in the software development life cycle and no software product can do without undergoing the process of maintenance. Estimating a software's maintainability effort and cost is not an easy task considering the various factors that influence the proposed measurement in software development. Abdulrahman et.al proposes an Evolutionary Neural Network (NN) model to predict software maintainability. The proposed model is based on a hybrid intelligent technique wherein a neural network is trained for prediction and a genetic algorithm (GA) implementation is used for evolving the neural network topology until an optimal topology is reached and the model was applied on a popular open source program, namely, Android. The results are very fine, where the correlation between actual and predicted points reaches 0.91 [37].

IV. CONCLUSION AND RESEARCH DIRECTIONS

In this paper, we show how GAs has been used in tackling many software engineering problems. The GAs has been used in various phases of software development like from requirement and analysis phase and software testing phase. It is also used developing new metrics. This will definitely help maturing software engineering discipline. There is urging to develop GAs based tools that become a part of software engineering and help in automating the software development process to optimal level. So there is a need for GAs community to come forward in help of software engineering discipline, so that full potential of GAs can be utilized in solving the problem faced by software professionals.Similar type studies must be carried out with large data sets to improve technique of test case generation. Moreover we can say that GAs is emerging field in software engineering.

REFERENCES

[1]. H. Seyed, M Hossein, and S Jalili, "SCI-GA: Software Component Identification using Genetic Algorithm”, Journal of Object Technology, 2013, pp. 1-3.

[2]. K Vijayalakshmi, N Ramaraj, and RAmuthakkannan, "Improvement of component selection process using genetic algorithm for component-based software development", International Journal of Information Systems and Change Management, 2008, pp. 63-80.

[3]. Y Singh, P K Bhatia, A Kaur, and O Sangwan, "Application of neural networks in software engineering: A review", In International Conference on Information Systems, Technology and Management, Springer Berlin Heidelberg, 2009, pp. 128-137.

[4]. M Harman, S AMansouri, and Y Zhang, "Search based software engineering: A comprehensive analysis and review of trends techniques and applications", Department of Computer Science, King’s College London, Tech. Rep. TR-09-03, 2009.

[5]. M R Girgis, "Automatic Test Data Generation for Data Flow Testing Using a Genetic Algorithm", J. UCS 11, 2005, PP.898-915.

[6]. G M Morris, D S Goodsell, R S. Halliday, Ruth Huey, William E Hart, R K Belew, and A J Olson, "Automated docking using a Lamarckian genetic algorithm and an empirical binding free energy function", Journal of computational chemistry 1998, pp.1639-1662.

[7]. H Mühlenbein, and D S Voosen, "Predictive models for the breeder genetic algorithm i. continuous parameter optimization", Evolutionary computation 1993, pp. 25-49.

[8]. J F Tang, L F Mu, C K Kwong, and X G. Luo, "An optimization model for software component selection under multiple applications development", European Journal of Operational Research 212, 2011, PP. 301-311.

[9]. J Pande, C J Garcia, and D Pan, "Optimal component selection for component based software development using pliability metric", ACM SIGSOFT Software Engineering Notes 38, no. 1, 2013, pp. 1-6.

[10]. A Dixit, and P. C. Saxena, "Software component retrieval using genetic algorithms", In Computer and Automation Engineering, 2009. ICCAE'09. International Conference on, IEEE, 2009, pp. 151-155.

[11]. S Parnami, K S Sharma, and S V Chande., "A survey on generation of test cases and test data using artificial intelligence techniques", International Journal of Advances in Computer Networks and its Security 2, no. 1, 2012, pp. 16-18.

[12]. A S Ghiduk, and M R Girgis, "Using genetic algorithms and dominance concepts for generating reduced test data", Informatica 34, no. 3 2010.

[13]. D C Koboldt, K M Steinberg, D E Larson, R K. Wilson, and E R. Mardis, "The next-generation sequencing revolution and its impact on genomics", Cell 155, no. 1, 2013, pp.27-38.

[14]. S M Mohi-Aldeen, R Mohamad, and S Deris, "Automatic Test Case Generation for Structural Testing Using Negative Selection Algorithm", 2009.

[15]. V Rajappa, ABiradar, and S Panda. "Efficient software test case generation using genetic algorithm based graph theory", In 2008 First International Conference on Emerging Trends in Engineering and Technology, IEEE, 2008, pp. 298-303.

[16]. Y Fuqing, M Hong, and LKeqin, "Software Reuse and Software Component Technology [J]", Acta Electronica Sinica 2, 1999.

[17]. S Sabharwal, R Sibal, and C Sharma, "Prioritization of test case scenarios derived from activity diagram using genetic algorithm", In Computer and Communication Technology (ICCCT), 2010 International Conference on, IEEE,2010, pp. 481-485.

[18]. R P Pargas, M J Harrold, and R R Peck, "Test-data generation using genetic algorithms", Software Testing Verification and Reliability 9, no. 4 1999, pp.263-282.

[19]. C Sharm, S Sabharwal, and R Sibal, "A survey on software testing techniques using genetic algorithm", 2014.
[20]. A Kaur, and S Goyal, "A genetic algorithm for regression test case prioritization using code coverage", International journal on computer science and engineering 3, no. 5, 2011, pp. 1839-1847.
[21]. R Krishnamoorthi, and SA S A Mary, "Regression test suite prioritization using genetic algorithms", International Journal of Hybrid Information Technology 2, no. 3, 2009, pp.35-52.
[22]. A Bertolino, "Software testing research: Achievements, challenges, dreams", In 2007 Future of Software Engineering, IEEE Computer Society, 2007, pp.85-103.
[23]. P McMinn, "Search-based software test data generation: A survey," Software Testing Verification and Reliability 14, no. 2, 2004, pp.105-156.
[24]. K Singh, "Effective Software Testing using Genetic Algorithms", Journal of Global Research in Computer Science 2, no. 4, 2011.
[25]. N Haghpanah, S Moaven, J Habibi, M Kargar, and S H Yeganeh, "Approximation algorithms for software component selection problem", In 14th Asia-Pacific Software Engineering Conference (APSEC'07), IEEE, 2007, pp. 159-166.
[26]. S Bhatia, A Bawa, and V K Attri, "A Review on Genetic algorithm to deal with Optimization of Parameters of Constructive Cost Model", International Journal of Advanced Research in Computer and Communication Engineering 4, no. 4, 2015.
[27]. B K Singh and A. K. Misra, "Software effort estimation by genetic algorithm tuned parameters of modified constructive cost model for nasa software projects", International Journal of Computer Applications 59, no. 9, 2012.
[28]. A Dhiman and C Diwaker ,"Optimization of COCOMO II effort estimation using genetic algorithm", American International Journal of Research in Science, Technology, Engineering & Mathematics 3, no. 2, 2013.
[29]. I Maleki, A Ghaffari and M Masdari, "A new approach for software cost estimation with hybrid genetic algorithm and ant colony optimization", International Journal of Innovation and Applied Studies 5, no. 1, 2014.
[30]. R Vankudoth, P Shireesha and T. Rajani, "A Model of System Software Components Using Genetic Algorithm and Techniques", International Journal of Advanced Research in Computer Science and Software Engineering, 2016, pp. 301-306.
[31]. A Martens, H Koziolek, S Becker, and R Reussner, "Automatically improve software architecture models for performance, reliability and cost using evolutionary algorithms", In Proceedings of the first joint WOSP/SIPEW international conference on Performance engineering, ACM, 2010, pp.105- 116.
[32]. J. A McCall, P. K, Richards and G. F. Wallers, "Factors in software quality ", Griffiths Air Force Base, N. Y: Rome Air Development Center Air Force Systems Command, 1977.
[33]. M Amoui, S Mirarab, S Ansari, and C Lucas, "A genetic algorithm approach to design evolution using design pattern transformation", International Journal of Information Technology and Intelligent Computing 1, no. 2, 2006, pp. 235-244.
[34]. D M Thakore and T Kamble, "Use of Genetic Algorithm in Quality Measurement", International Journal of Computer Applications 60, no. 8, 2012, pp. 24-28.
[35]. S H Aljahdali, and M E El-Telbany, "Genetic algorithms for optimizing ensemble of models in software reliability prediction", International Journal on Artificial Intelligence and Machine Learning (AIML) ICGST 8, no. 1, 2008, pp. 5-13.
[36]. AA B Baqais, M Alshayeb, and Z ABaig, "Hybrid intelligent model for software maintenance prediction", Proceedings of the World Congress on Engineering UK 2013.
[37]. M Jyoti, and L Bhambhu, "Modified Genetic Algorithm for Efficient Regression Test Cases", International Journal of Advanced Research in Computer and Communication Engineering, 2015, pp. 206-209.
[38]. R Malhotra and D Tiwari, "Development of a framework for test case prioritization using genetic algorithm", ACM SIGSOFT Software Engineering Notes 38, no. 3, 2013,pp.1-6.
[39]. S Yoo and M Harman, "Regression testing minimization, selection and prioritization: a survey", Software Testing, Verification and Reliability 22, no. 2, 2012, pp. 67-120.
[40]. R P Paragas, M J Harrold and R R Peck, "Test data generation using genetic algorithm", software testing verification & reliability 9, no. 4, 1994, pp.263-282.
[41]. P R Srivastava, and T Kim, "Application of genetic algorithm in software testing", International Journal of software Engineering and its Applications 3, no. 4, 2009, pp. 87-96.