Test Case Selection Using Bee Colony Optimization

Navdeep Koundal[1], Ankur Shukla[2], Mohsin Rashid Mir[3]

[1, 2, 3] School of Computer Science and Engineering, Galgotias University, Uttar Pradesh, India

Abstract: ***In software development life cycle (SDLC), testing phase is the most important phase. In regression testing there evolves the number of test cases that are impractical to test all of them. Therefore to overcome this problem, testing is done using selected test cases to reduce the testing effort and get the desired result accurately. We present the selection of test cases with the help of Bee Colony Optimization (BCO) Algorithm. The Algorithm proposed here uses swarm based intelligence technique on Maximum coverage to select the best of breed test case to evaluate any errors in the development cycle therefore the BCO produce optimal no. of test cases solving optimization problem.***

Keywords: Bee colony optimization, path testing, maximum coverage, regression testing**.**

1. Introduction

Testing is the process of evaluating a system or its component with the intent to find that whether it satisfies the specified requirements or not. In simple words testing is executing a system in order to identify any gaps, errors or missing requirements in contrary to the actual desire or requirements [1].

In software development life cycle Testing Phase is the important phase to develop the high quality software product. Software Testing is labor intensive process which also associated with high cost. As software once delivered comes in maintenance phase where it undergoes constant change and to test these changes we use Regression testing. Regression testing follows selective re testing techniques.

Test case selection/prioritization techniques try to find an ordering/ranking of test cases so that some test case adequacy can be maximized as early as possible [15]. Test case optimization and filtration depend on quality of initial population of test cases. Selection and prioritization of test cases are the two important solutions to the problem of test case optimization. Test case filtration and prioritization are closely related. In fact, test cases can be filtered by selecting the first N ordered test cases. Therefore, any test case prioritization algorithm can be used as a test case selection algorithm. Naturally, it is desirable to select those test cases that are most likely to reveal defects in the program under test.

Due to the resource and time constraints for re-executing large test suites, it is mandatory to optimize available test suites by using test cases prioritization, test case filtration, test case selection and test suite minimization. This paper is based on maximum coverage of path exist in problem program i.e. to find all the path execute in the program. In regression testing there is large number of test cases but due to time and cost constraint it’s not possible to execute all of them. So Test case optimization is done to reduce the number of test cases and selection is of effective test cases.

In Path testing there are number of Independent paths generated which are useful in testing the different type of coverage. Independent paths are generated with the help of Control flow graphs. We prioritize or select the test cases based on criteria or quality or fitness value of path generated by CFG. So researcher used the various approaches based on swarm intelligence which are search based for selection or prioritization of test cases. In the past number of Meta heuristic approaches have been used to solve the optimization problem but results are with serious demerits. The Bee Colony Optimization approach is capable to produce effective number of test cases which are able to maximum coverage of paths generated with the help of CFG .The effective test cases are those which covers all the nodes in minimum time of independent paths which are considered to be an important criterion to generate the effective test cases for path coverage.

2. Related Work

There are number of approaches have been proposed to reduces the no. of test cases by applying GA(Genetic algorithm) and ACO(Ant Colony Optimization). GA (Genetic Algorithm) is a directed search algorithms based on the “mechanics” of biological evolution which is developed by John Holland. GA Provide efficient, effective techniques for optimization used genetic operator to calculate fitness function. But There are some disadvantages of GA that it’s not so effective to solve a problem involving single measure for evaluation, external optimization, not sufficient to converge to a solution which cause results are to consistent.

The Ant Colony Optimization (ACO) meta-heuristic is a suite of algorithms inspired by the foraging behaviour of ants. Demerit of ACO is it waste large amount of computing resource to generate test cases. In ACO converges is guaranteed but time to convergence are uncertain, length of test sequence is higher and repetition of nodes in same sequence. The proposed algorithm is used to generate best number of test cases and to achieve maximum coverage of paths. Here we used BCO (Bee Colony Optimization) Algorithm where bees finding the food sites (test cases) then calculate their quality (fitness) which used to identify the best test cases with maximum coverage and less computing resources.

3. Bee Colony Optimisation: BCO-Approach

Bee colony optimisation (BCO) algorithm which is a swarm based meta-heuristic algorithm for optimizing problems. It was inspired by the intelligent foraging behaviour of honey bees .The model consists of three essential components: employed and unemployed foraging bees, and food sources. The first two components, employed and unemployed foraging bees, search for rich food Sites, which is the third component, close to their hive. The model also defines two leading modes of behaviour which are necessary for self-organizing and collective intelligence: recruitment of foragers to rich food sources resulting in positive feedback and abandonment of poor sources by foragers causing negative feedback [9].

In BCO, a colony of artificial forager bees (agents) search for rich artificial food sites (good solutions for a given problem). Then, the artificial bees randomly discover a population of initial solution vectors and then iteratively improve them by employing the strategies: moving towards better solutions by means of a neighbour search mechanism while abandoning poor solutions.

4. Proposed Strategy

The Proposed strategy used to solve the problem of Test Case Optimization using BCO Algorithm [12]. The aim of proposed approach is to find the best food source (Test case with maximum coverage). The positions of food sources represent the solutions of the optimization problem and the amount of food which defines the fitness or quality of the solution. The Main steps of BCO Algorithms are:

- Place the Employed bees on food sources and measure their amount of food.
- Calculate the probability value of food source with which they are selected by Selector bee.
- Stop the exploitation process of food source unused by the bees.
- Sent the scout bees for discovering new food source to search area randomly.
- Remember the best food source found so far.

The entire process of working is divided in between three types of bees. They are:

- Employed bee
- Onlooker Bee
- Scout bee

The main Principle and working of proposed approach:
BCO approach is a Meta heuristic population based approach. Solution of optimisation problem is represented by each test case. The quality of each test case is calculated by the fitness value of problem. The work proposes that by using the algorithm we generate the optimise test cases and will contain all possible independent paths along with its test data. The test data will be the required input to be given to the program, for travelling along the path and vice versa [6].

Initially we make the program corresponding Control Flow Graph. Now from control flow graph, the different independent paths are generated. Each independent path would comprise number of normal nodes and predicate nodes. Every independent path would represent a Test Case .Now the BCO algorithm is applied to generate a test suite of selected test cases which would passes through the independent paths and hence into to the test cases.

Consider the figure 2. Here the scouts' bee acts as search agent which searches for the execution state of the test cases also initialises the test cases with the initial test data with the help of white box testing techniques. Then the bee calculates the fitness value of each test case by computing the coverage of each node. This process is repeated till we found executable state.

Then the employee's bee passes the fitness value of the processed nodes to the onlooker agent. The employee bee compares the fitness value of nodes with the fitness value of the neighbouring nodes. If the fitness value of node is found greater than the neighbouring fitness value, then the node's information is stored in the optimal test case database. The node whose fitness value is found less is stored in the abandoned repository. New Test data are again generated by scouts' bees from the abandoned repository and again the same process of employed bee is repeated. The algorithm for test case optimization using Bee Colony optimization approach is presented in the further section.

5. Algorithmic Structure of BCO

The general algorithmic structure of the BCO optimization approach and pseudo code is given as follows:

```
Initialization Phase
REPEAT
Employed Bees Phase
Onlooker Bees Phase
Scout Bees Phase
Memorize the best solution achieved so far

UNTIL (Cycle = Maximum Cycle Number or a Maximum CPU time)
```

1. In the initialization phase, the population of food sites (solutions) is initialized by artificial scout bees and control parameters are set.
2. Search for an executable state and evaluate the test node.
3. Initialise the current path as cycle=1
4. Repeat
5. Produce initial food sites randomly that correspond to solution using:
   $X_{ij}= Xj^{min} + \text{rand} (Xj^{max}-Xj^{min})$
   Rand (0, 1)
6. Test data are generated with the help of white box testing techniques like equivalence partitioning and boundary value analysis.
7. Greedy selection process is applied on generated test data.
8. Pass the generated test data to SUT and calculate the fitness value.
9. Test cases with highest fitness value are selected by onlookers' bee and leave the rest.

10. Same process is repeated till a particular test data with 100% fitness value and 0% fitness value is produced.
    $P_{m=} fit_m (X_m)/\sum fit_m (X_m)$
    Where P_m is the probability function which signifies the probability with which the ith test data traversers an independent test path successfully.
11. Store the test case to the optimal database.
12. New Test data generated by scouts bee in next iteration and go to step 5

For implementation of the above algorithm, our approach uses the BCO Test Case Optimization tool to select the best Test Cases with maximum coverage by applying the BCO algorithm. The tool considers a program as an input to generate independent paths. Using the generated independent paths Test Cases are traversed along the paths with the help of ABC algorithm. On executing this test cases with maximum coverage (High fitness Value) are selected. Finally the optimal Test cases are generated.

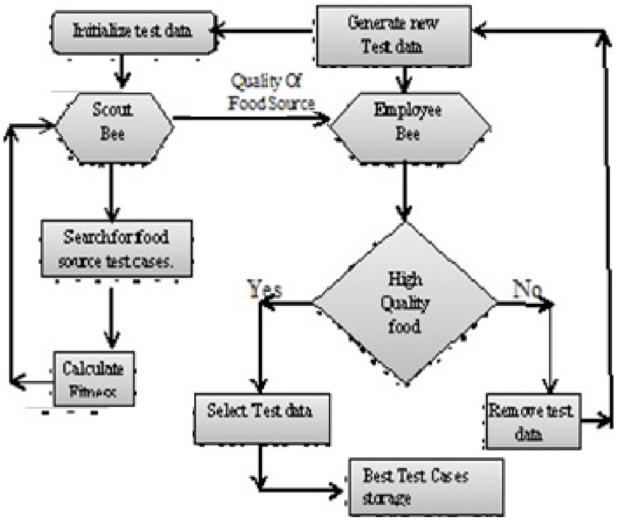


Figure1: Flow Diagram BCO algorithm

6. Problem Solved Using BCO Based on Maximum Coverage

Test Case Prioritization based on white box testing helps to prioritize the test cases based on the internal structure of the program. Therefore knowledge of internal structure of code used to find the test cases required to provide maximum test coverage. [4] Path testing is a type of white box testing that selects the set of test paths exist in the program. In path testing it picks number of paths to assure that every program statement is executed at least once.

6.1 Example

For illustration purpose we have taken the triangle problem to classify a triangle. Inputs are given to three sides of triangle and out will be one of these: scalene, isosceles, equilateral and not a triangle [1]. On executing the program code provided as input to the tool. The forager bees i.e. onlooker bees will generate different numbers of paths. In the program there are 18 nodes and number of independent path generated are:

1 ABFGNPQR
2. ABFGNOQR
3. ABCEGNPQR
4. ABCEQNOQR
5. ABFGHJIMQR
6. ABFGHJKMQR
7. ABFGHIMQR

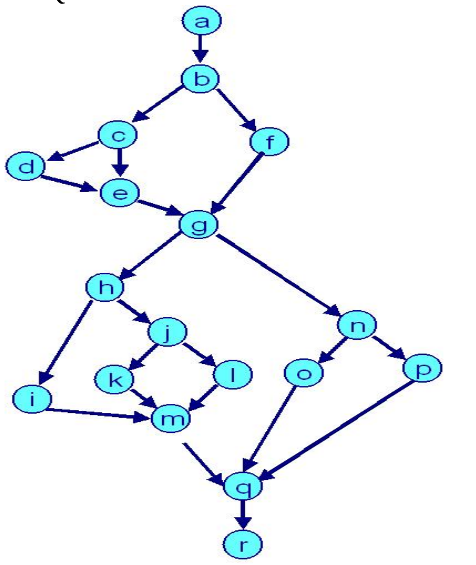


Figure 2: Flow graph of the triangle problem code

According to proposed BCO algorithm, the steps takes places are:
1. All the paths are initializes as test cases.
2. Scout bees Initialize the current traversal path as cycle=1
3. Repeat
4. Generate new test cases X_{ij} using formula for movement of scout bees:
   $X_{ij}= Xj^{min} + rand (Xj^{max}-Xj^{min})$
   X_{ij}= initial test case
   Xj^{min} (minimum numbers of test cases) =1
   Xj^{max} (maximum numbers of test cases) =7
   Rand=A random Number (0, 1)
   So, Xij= 1 + 0 * 7-1 = 1 (Path number)
5. ABFGNPQR is selected by employee bee as it traverses.
6. Generate the test data using functional testing techniques.
7. Calculate the fitness value.
8. Onlooker bees selects the test cases which having highest fitness value and leave the rest.
9. In the next iteration cycle scouts bee generate the new test data and repeat step 5.
10. Selected test cases or data are stored in database.
11. Go to Step 1 and Repeat the same the process for other test paths.

Table 1: Test cases with input, output path covered and fitness value

S No	Test Case	Output	Path Covered	Fitness Value %
1	50-50-1	Isosceles	ABCDEGHJKMQR	100
1	50-50-2	Isosceles	ABCDEGHJKMQR	100
1	50-50-99	Isosceles	ABCDEGHJKMQR	0
2	50-50-50	Equilateral	ABCDEGHIMQR	100
2	80-50-80	Isosceles	ABCDEGHIMQR	0
3	50-1-50	Isosceles	ABCDEGHJKMQR	100
3	50-2-50	Isosceles	ABCDEGHJKMQR	0
3	1-50-50	Isosceles	ABCDEGHJKMQR	0
4	50-50-100	Not a Triangle	ABCDEGNOQR	0
4	100-50-50	Not a Triangle	ABCDEGNOQR	100

If the test case data covered all the nodes of path so the fitness value with respect to path is 100% where if they don't covered test node completely then for the path ABFGNPQR, fitness value are 0%.

7. Comparison

Using the swarm based intelligence algorithm there are exist a probability of falling into a Local optimum but using Bee colony optimization algorithm probability is low. As BCO work as combination of both local and global search ability with main aim is to improve. Here is some figure which shows that using BCO approach it achieves maximum coverage based on Number of Cycles, percentages of path covered which increases and maintain consistency after a fix threshold value.

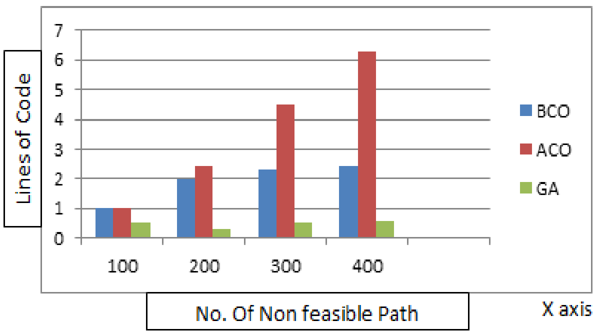


Figure 3: No of lines of code vs. non-feasible path

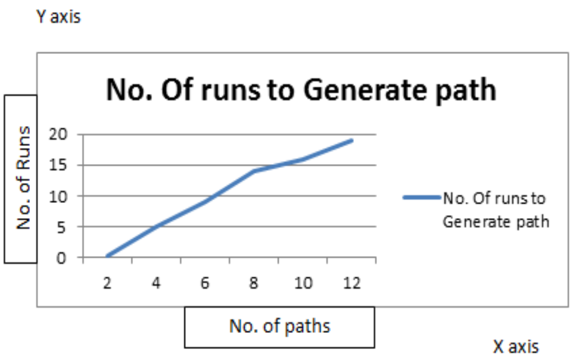


Figure 4: No of runs vs. paths

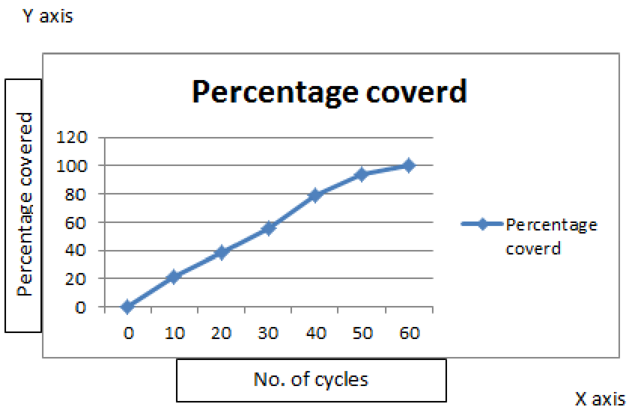
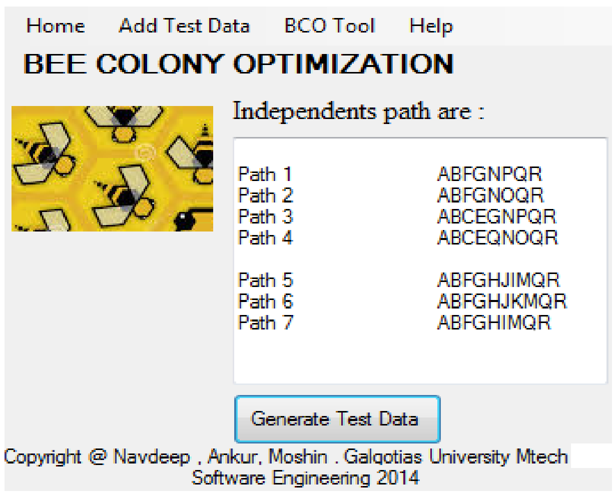


Figure 5: No of cycles vs. percentage covered

The graph from Figure 4 plots the number of runs and path showing the paths which increased incrementally. The line of graph is having lesser slope. The Figure 5, graph plots comparison between BCO, ACO and GA [2] between number of non-feasible paths and the LOC. This graph testifies that the BCO distinguishes and accounts less number of non-feasible paths. Thus more number of feasible paths with test data is covered. But GA doesn't find any non-feasible paths. Non-feasible paths are the paths which cannot be processed using any generated test data. Hence, the non-feasible path is technically accessible but logically cannot be processed. This approach of BCO will return non feasible paths if any path generated using BCO are not accessible using any test data.

8. Screenshots

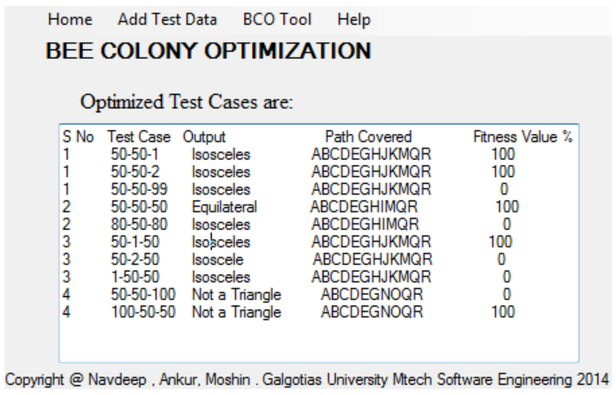


BEE COLONY OPTIMIZATION

Independents path are :

Path 1	ABFGNPQR
Path 2	ABFGNOQR
Path 3	ABCEGNPQR
Path 4	ABCEQNOQR
Path 5	ABFGHJIMQR
Path 6	ABFGHJKMQR
Path 7	ABFGHIMQR

Copyright @ Navdeep , Ankur, Moshin . Galgotias University Mtech Software Engineering 2014



BEE COLONY OPTIMIZATION

Optimized Test Cases are:

S No	Test Case	Output	Path Covered	Fitness Value %
1	50-50-1	Isosceles	ABCDEGHJKMQR	100
1	50-50-2	Isosceles	ABCDEGHJKMQR	100
1	50-50-99	Isosceles	ABCDEGHJKMQR	0
2	50-50-50	Equilateral	ABCDEGHIMQR	100
2	80-50-80	Isosceles	ABCDEGHIMQR	0
3	50-1-50	Isosceles	ABCDEGHJKMQR	100
3	50-2-50	Isoscele	ABCDEGHJKMQR	0
3	1-50-50	Isosceles	ABCDEGHJKMQR	0
4	50-50-100	Not a Triangle	ABCDEGNOQR	0
4	100-50-50	Not a Triangle	ABCDEGNOQR	100

Copyright @ Navdeep , Ankur, Moshin . Galgotias University Mtech Software Engineering 2014

9. Conclusions and Future Work

In this paper, the test case selection is done based upon Bee Colony Optimization (BCO).We investigated the performance of standard of the Bee Colony algorithm and compared their performances against other swarm based intelligence algorithm. Also the results' showing that BCO approach is much better than the other optimizing algorithm. BCO approach is more efficient and work faster for optimizing test cases. The proposed BCO algorithm has been explained using example on maximum coverage or maximum path coverage. The search bee looks for path until they get 100% coverage test data. In future, there different other factors can be added to increase the efficiency of the BCO approach and give more prioritized and optimized test case results.

References

[1] Aggarwal, K. K.; Singh, Y. (2001): A book on software engineering. New Age International (P) Ltd.; Publishers, 4835/24, Ansari Road, Daryaganj, New Delhi.
[2] AL-Salami, N.M.A. (2009): Evolutionary Algorithm Definition, American J. of Engineering and Applied, Science Publications, pp.789-795.
[3] Alspaughy, S.; Walcotty, K. R.; Belanichz, M.; Kapfhammerz, G. M.; Soffa, M. L. (2007): Efficient Time-Aware Prioritization with Knapsack Solvers, Proceedings of the ASE 2007 Workshop on Empirical Assessment of Software Engineering Languages and Technologies. Atlanta, Georgia, pp. 1-6.
[4] Askarunisa, A.; Shanmugapriya, L.; Ramaraj, N. (2009): Cost and Coverage Metrics for Measuring the Effectiveness of Test Case Prioritization Techniques, INFOCOMP Journal of Computer Science, pp. 1-10.
[5] Byson, N. (1995): A goal programming method for generating priorities vectors, Journal of Operational Research Society, Palgrave Macmillan Ltd.,Houndmills, Basingstoke, Hampshire, RG21 6XS, England, pp. 641-648.
[6] Chong, C. S. et. al. (2006): A Bee Colony Optimization Algorithm to Job Shop Scheduling, Journal Winter Simulation Conference, Monterey, CA, pp. 1954-1961.
[7] Crawford, G.; Williams, C. (1985): A note on the analysis of subjective judgment matrices, Journal of Mathematical Psychology, The Rand Corporation, USA, pp. 387-405.
[8] Li, Z.; Harman, M.; Hierons, R.M. (2007): Search algorithms for regression test case prioritization, IEEE Transactions on Software Engineering, San Francisco, CA, USA, pp. 225-237.
[9] Mala, D. J.; Mohan, V. (2009): ABC Tester - Artificial Bee Colony Based Software Test Suite Optimization Approach, International Journal of Software Engineering, Sprinter Global Publication, pp. 15-43.
[10] McCaffrey, J. D. (2009): Generation of pair wise test sets using a simulated Bee Colony Algorithm, 10th IEEE International Conference, IEEE Press Piscataway, NJ, USA, pp. 115-119.
[11] Navrat, P.; Jelinek, T.; Jastrzembska, L. (2009): Bee hive at work: A problem solving, optimizing mechanism, In Proceedings of Nature & Biologically Inspired Computing, IEEE Conferences, Coimbatore, pp. 122-127.
[12] Owaied, H.; Saab, S. (2008): Modeling Artificial Bees Colony System, ICAI, Proceedings of the 2008 International Conference onArtificial Intelligence, Las Vegas, Nevada, USA, pp.443-446.
[13] Pacurib, J. A.; Seno, G.M.M.; Yusiong, J.P.T. (2009): Solving Sudoku Puzzles Using Improved Artificial Bee Colony Algorithm, Fourth International Conference on Innovative Computing, Information and Control, Kaohsiung, Taiwan, pp. 885-888.
[14] Rahmatizadeh, Sh.; Shah-Hosseini, H.; Torkaman, H. (2009): The ant-bee routing algorithm: A new Agent based Nature-Inspired Routing Algorithm, Journal of Applied Sciences, Addison Wesley Publishers, Harlow. Esses. UK, pp. 983-987.
[15] Rothermel, G. Untch, R.H.; Chu, C.; Harrold, M.J. (1999): Test Case Prioritization: An Empirical Study, In Proceedings of the International Conference on Software Maintenance, Oxford, UK, pp. 179-188.
[16] Wong, L.P.; low, M. Y. H.; Chong, C. S. (2008): A Bee Colony Optimization Algorithm for Travelling Salesman Problem, Second
[17] Asia International Conference on Modeling & Simulation, IEEE Computer Society, Washington, DC, USA, pp. 818-823.
[18] Yang, X.S. (2005): Engineering Optimizations via Nature-Inspired Virtual Bee Algorithms, Artificial Intelligence and Knowledge Engineering Applications: A Bio Inspired Approach, Lecture notes in computer science, Springer Berlin / Heidelberg, pp. 317-323.

Author Profile

Navdeep Koundal pursuing M. Tech. in Software Engineering from Galgotias University. He has completed B.Tech. in Information and Technology from Himachal Pradesh University Shimla in 2010 Area of interest is Software Testing, Software Engineering.

Ankur Shukla pursuing M. Tech. in Software Engineering from Galgotias University. He has completed B.E. in Computer Science and Engineering from Guru Ram Das Khalsa Institute Of Science and Technology Jabalpur M.P. in 2010 Area of interest is Software Testing, Software Quality & Estimation.

Mohsin Rashid Mir pursuing M. Tech. in Software Engineering from Galgotias University. He has completed B.Tech. in Information and Technology from Maharishi Dayanand University Rohtak Haryana India in 2011 Area of interest is Software Testing, Software Engineering .methodologies.