Search-based Software Requirements Selection: A Case Study

A. Charan Kumari
Department of Physics & Computer Science
Dayalbagh Educational Institute
Dayalbagh, Agra, India

K. Srinivas
Department of Electrical Engineering
Dayalbagh Educational Institute
Dayalbagh, Agra, India

ABSTRACT
This paper presents a Multi-objective Quantum-inspired Hybrid Differential Evolution (MQHDE) for the solution of software requirements selection problem and its application on a real-world project. As the customer requirements change from time to time, often software products are developed in an iterative or incremental manner so as to deal with these changing requirements. The problem is to identify a set of requirements to be included in the next release of the software product, by minimizing the cost and maximizing the customer satisfaction. This problem is referred to as Multi-objective Next Release Problem (MONRP) in the jargon of Search-based Software Engineering (SBSE). The solution to the problem of MONRP has been studied by researchers using different metaheuristic search techniques. The efficiency of the proposed MQHDE is tested on a real-world application and the results are compared against the state-of-the-art multi-objective evolutionary algorithm NSGA-II, and found that the performance of MQHDE is promising and therefore can be used with confidence for the solution of real-world instances of MONRP.

General Terms
Software Engineering, Requirements Engineering.

Keywords
Search-based software engineering, Multi-objective optimization, Multi-objective next release problem.

1. INTRODUCTION
Search-based Software Engineering (SBSE) has emerged as a promising research field. This recent discipline involves the modeling and resolution of complex software engineering problems as optimization problems, especially with the use of metaheuristics [1]. The term Search-based software engineering was coined by Mark Harman and Bryan Jones in 2001 [1], and provided an insight into the application of the metaheuristic search techniques to solve different problems in the software engineering. Since then the researchers applied the search techniques in different phases of the software development life cycle starting from requirements engineering[2], project planning and cost estimation[3], through design[4], testing [5,6] and to maintenance[7]. Most of the problems in SBSE are NP-hard and hence cannot be solved efficiently by traditional optimization techniques especially for the large problem instances. Therefore, metaheuristic search techniques are used for the solution of these problems. Though, Metaheuristic search techniques do not guarantee to provide optimal solutions, yet, they can obtain near-optimal solutions in a reasonable amount of computational time. SBSE field is gaining popularity due to its ability in handing the complex and large problem instances.

In order to deal with the changing requirements of the customers from time to time, often software products are developed in an iterative or incremental manner. Companies involved in developing and maintaining large complex software systems require to determine the requirements of the customers to be included into its next release. This problem has been formulated as Next Release Problem (NRP) by Bagnall *et al.* [2] and is widely referenced by researchers in the field of search-based software engineering. The problem is defined as to identify a set of requirements to be included into the next release of the software product, by satisfying the demands of the customers to the maximum extent, and at the same time ensuring the minimum utilization of the resources as far as possible. The goal of NRP is to balance customer satisfaction, resource constraints and requirement dependencies. In his paper he applied various techniques including Greedy algorithms and simulated annealing on a set of five randomly generated data sets with increasing dimensionality of the problem.

Greer and Ruhe [8] proposed a Genetic algorithm based approach for software release planning in an incremental manner. In their paper they presented a new method called EVOLVE for software releasing planning. Given a set of requirements with their effort estimations and the grouping of these requirements into priorities by the customers, the method uses a Genetic Algorithm to derive potential release plans with in the specified technical constraints. They studied the applicability of the approach on a sample software project.

To model a more realistic and real life application, the NRP has been formulated by Zhang *et al.* [9] as Multi-Objective Next Release Problem (MONRP). As the objectives in real applications are contradictory in nature, in his work he formulated the problem as multi-objective and defined it as the selection of candidate requirements for their implementation in the next release of the software product, by minimizing the cost of implementing the requirements in terms of money, resources, time etc. and at the same time maximizing the customer satisfaction by including these requirements. Later Durillo.J. *et al.*[10] studied MONRP by applying NSGA-II, Pareto GA, Single-objective GA and Random Search on a set of six randomly generated data sets and experimented with the scaling and boundary issues of the MONRP.

Many researchers studied the MONRP from different perspectives. Anthony Finkelstein *et al.* [11] introduced the concept of fairness in requirements optimization using a new formulation of MONRP by setting up three fairness models to balance the requirements fulfillments among different customers. They applied NSGA-II, The Two-Archive algorithm and random search on two real world data sets and reported the results. Sensitivity analysis in Requirement Engineering is studied by Mark Harman *et al.* [12] to identify requirements that are sensitive to inaccurate cost estimations. After performing an empirical study on synthetic and real world data set, they confirmed the assumption through statistical analysis that, more expensive requirements and higher level of inaccuracies tend to have greater impact on NRP. The problem of balancing the requirements of today with those of future, coined as Today/Future Importance analysis (T/FIA) was addressed by Zhang *et al.* [13]. They considered the problem of finding a suitable set of requirements that balances the needs of today against the needs for future. In their work three objective functions are defined - to maximize the customer satisfaction for today, for the future and to minimize the implementation cost. They reported the results with this formulation on synthetic as well real world data set.

The problem of MONRP is also solved using the recent trends in evolutionary algorithms like quantum computing. Charan Kumari *et al.* proposed a Quantum-inspired Elitist Multi-objective Evolutionary Algorithm (QEMEA) [14] and Multi-objective Quantum-inspired Hybrid Differential Evolution (MQHDE) [15] for the solution of MONRP and studied its performance on six benchmark problems derived from the literature and found the results to be consistent and superior to the results reported in the literature.

The basic model of MONRP is considered in this work due to its wide applicability in all software companies and manufacturing companies. As the solution of MONRP problem is a supportive aid for the software engineers in decision making during requirements engineering phase, the data pertaining to a banking sector application project of an International software development company is collected and studied as a case study.

The rest of the paper is structured as follows. The formulation of software requirements selection process as Multi-Objective Next release Problem is presented in Section 2. The pseudo code of MQHDE along with the detailed description of the concepts is given in Section 3. Section 4 discusses the efficacy of the proposed algorithm on a real-world application project and reports the results. Finally, concluding remarks are given in Section 5.

2. MULTI-OBJECTIVE NEXT RELEASE PROBLEM

This section explains the Multi-objective Next Release Problem for the selection of software requirements as devised by Zhang *et al.* [9].

Assuming that for an existing software, there are 'm' customers $\{c_1, c_2, c_3, .. c_m\}$ who proposed 'n' requirements $\{ r_1, r_2, r_3, \ldots r_n\}$, to be considered for the inclusion into next release of the product. Based on the importance of the customer, the company assigns a weight factor. Let the set of relative weights associated with each customer c_j $(1 \le j \le m)$ is denoted by Weight = $\{ w_1, w_2, w_3, \ldots w_m \}$, where $w_j \in [0,1]$ and $\sum_{j=1}^{m} w_j = 1.$

Let the cost associated with each requirement ri ($1 \le i \le n$) for its implementation is designated by Cost = $\{ cost_1, cost_2, cost_3, \ldots cost_n \}$. As the requirements priority differs from customer to customer, each customer c_j ($1 \le j \le m$) assigns a priority value for each of their requirement r_i ($1 \le i \le n$), denoted by value (r_i,c_j).
The score of requirement r_i can be calculated as

$$score_i = \sum_{j=1}^{m} w_j * values(r_i, c_j) \quad (1)$$

The decision vector x = $\{x_1, x_2, x_3, \ldots x_n \} \in \{0,1\}$ is a solution vector that indicates the requirements that are to be included in the next release of the software product. The decision variable x_i takes a value 1, if the requirement is selected for the next release and a 0, otherwise.

The two objectives to optimize can be formulated as

$$\text{minimize } f_1 = \sum_{i=1}^{n} \cos t_i * x_i \quad (2)$$

$$\text{maximize } f_2 = \sum_{i=1}^{n} score_i * x_i \quad (3)$$

The main goal in implementing the MONRP is to find a set of requirements that are to be included in the next release of the software product by minimizing the cost and at the same time maximizing the customer satisfaction.

3. Multi-objective Quantum-inspired Hybrid Differential Evolution

Differential Evolution (DE) is a stochastic, population based evolutionary algorithm [16] which has been widely applied for solving multi-dimensional global optimization problems in various fields. But, the design of its mutation and crossover operators makes it inapt for many combinatorial optimization problems over binary space. On the other hand, on some attempted problems Quantum-inspired Evolutionary Algorithms (QEAs) [17] performed better than evolutionary techniques due to their ability to balance exploration and exploitation of the search space, robust search and efficiency in representation scheme. In this section we present a state-of-the-art elitist Multi-objective Quantum-inspired Hybrid Differential Evolution algorithm (MQHDE) [15] which is designed not only to make DE suitable for effectively handling combinatorial optimization problems but it also effectively combines the strengths of Differential Evolution, Genetic Algorithm and the principles of Quantum Computing and extends it to Multi-objective optimization framework for the solution of Multi-objective Next Release Problem. The other main features of MQHDE are as follows:

- It uses an efficient and powerful quantum representation. This representation provides probabilistically a linear superposition of multiple states and thereby brings an additional element of randomness and new dimension into the algorithm. It reduces the population size, improves the search capability and even speedup the convergence.
- It strikes the right balance between exploration of the search space and exploitation of the reached good regions in the search space.
- It also effectively hybridizes Differential Evolution and Genetic Algorithm to make the search more effective. In every generation 50% of the Quantum population is obtained using a Quantum Mutation operator specially designed for multi-objective framework to guide the individuals towards better solutions (i.e. which increase proximity to the Pareto-optimal front). The designed mutation operator is adaptive and quite versatile making the approach almost parameter free and easy to use unlike Quantum-inspired Evolutionary algorithms, where the use of quantum rotational gates require careful

designing of the lookup table for the rotational angle and the direction for updating the quantum individuals which is problem specific and improper design of lookup table may lead to poor performance of the algorithm. The remaining quantum population is obtained using Quantum Uniform crossover.

- Maintains sufficient diversity in the population.
- Use of Elitism ensures that the quality of solutions in the population does not deteriorate and this is further strengthened by the used representation and the quantum mutation and crossover operators.
- Use of time tested Fast Non-domination Sorting [18].
- Does not require the maintenance of external archive for storing non-dominated solutions.
- Scalability to more number of objectives
- Consistent performance in terms of Quality Pareto solutions with good spread, more number of solutions in the obtained front and Faster convergence.

This section describes the framework of the MQHDE. The pseudo code of MQHDE is presented in Algorithm1.

3.1 Representation

MQHDE employs a probabilistic representation that is based on the concepts of qubits. It maintains a quantum population $Q = (q_1,q_2,q_3,\ldots q_n)$ of size 'n', where q_i is a qubit individual defined as

$$q_i = (\gamma_{i1}, \gamma_{i2}, \gamma_{i3}, \ldots \gamma_{im}) \quad , \qquad (4)$$

where m is the length of the qubit individual. The probability that qubit is in state '0' is represented by $\left|\gamma_{ij}\right|^2$ and in state '1' by $\left|1-\left|\gamma_{ij}\right|\right|^2$. Each γ_{ij} is initialized in the range [-1, 1].

Each qubit individual q_i is observed to make a binary solution P_i , using the following pseudo code.

```
procedure  Observe(q)
begin
        j = 1
        while (j < m)
        {
           if (rand[0,1] < |γij|² )
               pij = 0
           else
               pij = 1
           end;
           j = j + 1
        }
end
```

3.2 Mutation operator

The Mutation operator of DE is applied on qubit individuals and is represented as [19]

$$q_i(t) = q_{elite}(t\text{-}1) + F * (q_{r1}(t\text{-}1) - q_{r2}(t\text{-}1)) \qquad (5)$$

where *r1* and *r2* are two random numbers, and are distinct and also different from the running index *i*. The parameter *t* denotes the generation number. q_{elite} is an elite picked up randomly from the elites belonging to the best non-dominated front obtained in the previous quantum population. F is the mutation control parameter and is generally set in the range of [0, 2]. The designed mutation operator is adaptive and quite versatile making the approach almost parameter free and easy to use.

Algorithm1: Pseudo code of MQHDE

1: t = 0
2: Initialize Q(t) a population of 'N' qubit individuals with 'm' qubits in each.
3: Obtain P(t) by observing the states of Q(t).
4: Evaluate fitness of P(t).
5: Perform fast non-dominated sort on P(t)
6: **while** not termination condition **do**
7: t = t +1
8: Obtain 50% of the offspring population Q(t) using the quantum mutation operator applied on parent population Q(t-1) and elites of Q(t-1) as shown below:
$q_i(t) = q_{elite}(t\text{-}1) + F * (q_{r1}(t\text{-}1) - q_{r2}(t\text{-}1))$,
where $r_1 \neq r_2 \neq i$ and $F \in [0,2]$.
9: Obtain the remaining offspring population of Q(t) using Quantum uniform crossover.
10: Obtain P(t) by observing the states of Q(t).
11: Evaluate the fitness of P(t).
12: Perform fast non-dominated sort on P(t-1) U P(t)
13: Form Q(t) by accommodating distinct quantum individuals pertaining to the different Pareto-fronts starting from the best front by taking crowding distance into consideration.

3.3 Crossover Operator

The crossover operation operates on the original Quantum individuals. In MQHDE, Uniform crossover is used with a crossover probability of 0.8. In this process, each offspring quantum individual is generated by randomly selecting each quantum gene from either of the two selected quantum individual parents.

3.4 Selection

The qubit individuals of the next generation are selected after performing a fast non-dominated sorting [18] on the population obtained by combining parent and offspring populations. In this process, for each solution two entities are computed - domination count n_p, the number of solutions which dominate the solution p and S_p, a set of solutions that the solution p dominates. Hence, all solutions in the first nondominated front will have their domination count as zero. Then, for each solution p with $n_p = 0$, the domination count of each member (q) of its set S_p is decremented by one. Then the second nondominated front is identified as all q for which the domination count becomes zero. The process is repeated until all the fronts are identified.

The quantum population Q(t) for the next generation is obtained by accommodating the quantum individuals belonging to various Pareto-fronts starting from the first front. In case, the number of quantum individuals present in the considered Pareto front is less than or equal to the number of vacant slots, then all the solutions of the front are accommodated. Otherwise, based on the available slots quantum individuals are selected by taking crowding distance measure into consideration, to ensure diversity.

4. CASE STUDY: PERFORMANCE EVALUATION OF MQHDE ON A REAL-WORLD INSTANCE OF MONRP

This section presents the solution of real world instance of MONRP problem concerning a banking project using MQHDE. This data is provided by a large multinational software development company but the identity of the company and the project details cannot be revealed due to the company's policies and privacy issues related to an ongoing project. In this section the performance of MQHDE is examined with respect to the well-known state-of-the-art multi-objective evolutionary algorithm NSGA-II. The description of the problem and the other details are presented in the beginning of this section followed by the results obtained by the two algorithms. The performance of the algorithms is evaluated based on different metrics, after performing 100 independent runs. An in-depth analysis of the obtained results is presented towards the end of this section.

4.1 Problem Description

The data set consists of 10 customers and 60 requirements. Each customer is ranked with some weight factor on the scale of 1.0 to 10.0; revealing the importance of the customer to the company. The cost of implementing each requirement is provided in terms of GBP. Each customer assigned a grade to each requirement on a scale of 0 to 5 indicating the priority of the requirement.

4.2 Algorithmic Parameters

The population size is taken as 100. The performance measures are calculated after 10000 function evaluations. In NSGA-II, tournament selection is used for the selection of parents. The genetic operators used are - Single point crossover with a crossover probability of 0.9 and bitwise random mutation with a mutation probability of 0.1. In MQHDE, the mutation control parameter F is randomly initialized between 0 and 2, and a uniform crossover with a crossover probability of 0.8 is considered.

4.3 Metrics for Performance Evaluation

All comparisons are based on qualitative and quantitative measures. Qualitative comparison is based on the plots of the final Pareto fronts obtained. For assessing the quality of the results obtained by the multi-objective algorithms two issues are generally taken into consideration: (i) convergence towards the Pareto-optimal front (ii) maintaining the diversity in the obtained solutions. In this paper, we have considered three metrics – (a) Generational Distance (GD) to measure closeness of the obtained solutions to the Pareto optimal front (b) Spread (Δ) to measure the diversity among the obtained solutions and (c) Hypervolume (HV) for measuring both closeness and diversity.

In order to compute Generational distance and Spread metrics, a Pareto optimal front is required. As Pareto optimal front is not known in this case, a reference Pareto optimal front is constructed by collecting the non-dominated solutions obtained during 50 independent runs of the MQHDE and NSGA-II.

- **Metric for Convergence**

The Generational Distance (GD) [20] metric calculates the average distance of the obtained solutions from the Pareto optimal front as

$$GD = \frac{\left(\sum_{i=1}^{|Q|} d_i^2\right)^{1/2}}{|Q|}, \qquad (6)$$

Where d_i is the Euclidean distance between the solution i in the obtained front and the nearest member of Pareto optimal front and Q represents the number of solutions in the obtained front. This metric is applied after normalizing the objective function values. For an ideal distribution, this metric takes a value zero.

- **Metric for Diversity**

The spread metric Δ [20] measures the extent of spread by the obtained solutions

$$\Delta = \frac{\sum_{m=1}^{M} d_m^e + \sum_{i=1}^{|Q-1|} |d_i - \bar{d}|}{\sum_{m=1}^{M} d_m^e + |Q-1|\bar{d}}, \qquad (7)$$

where d_i can be any distance measure between consecutive solutions and $\bar{d}$ is the mean value of these distance measures. The parameter d_m^e is the distance between the extreme solutions of obtained front and Pareto optimal front corresponding to the m-th objective function. M corresponds to the number of objective functions and Q represents the number of solutions in the obtained front. This metric is applied after normalizing the objective function values. This metric takes a value of zero, for an ideal distribution,

- **Metric for Convergence and Diversity**

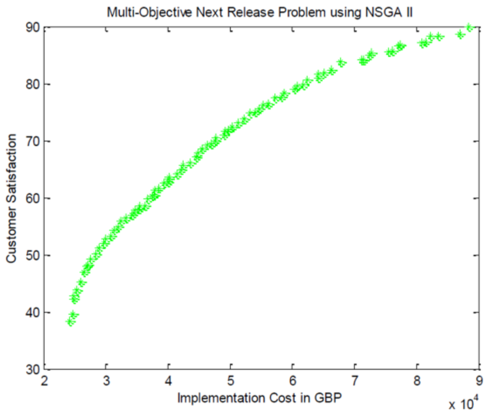
The Hypervolume (HV) [20] metric calculates the volume covered by the members of the obtained front in the objective space. Mathematically, for each solution $i \in Q$ (number of solutions in the obtained front), a hypercube v_i is constructed with a reference point w and the solution i as the diagonal corners of the hypercube. The reference point is found by taking the maximum possible objective values. And as the objective function values are of differing magnitude, the objective function values are normalized and (1, 1) is taken as the reference point. A union of all the hyper cubes is found and its hypervolume is calculated.

$$HV = volume\left(\bigcup_{i=1}^{|Q|} v_i\right), \qquad (8)$$

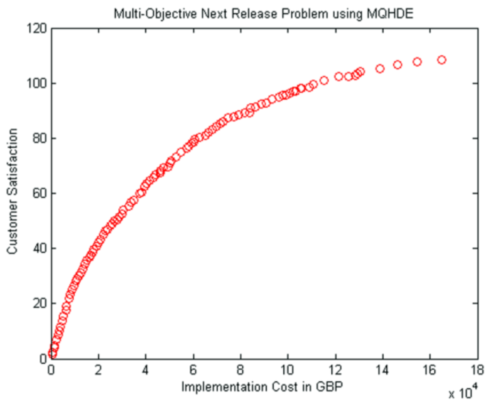
The statistical results are collected after performing 100 independent runs, and the results are tabulated and also illustrated by statistical box plots.

Table 1. Results of performance metrics

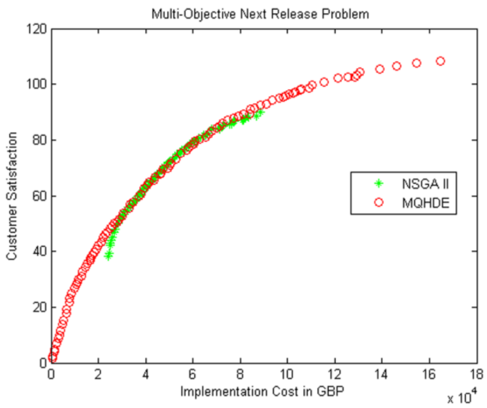
	NSGA-II		MQHDE	
	Mean	Std.	Mean	Std.
Generational Distance	0.001650	0.000615	0.000624	0.000153
Spread	0.794417	0.043972	0.369149	0.030922
Hypervolume	0.662813	0.024792	0.722523	0.009454



(a). Pareto front obtained by NSGA-II



(b). Pareto front obtained by MQHDE



(c). Pareto front obtained by NSGA-II & MQHDE

Figure 1 : Pareto fronts

4.4 Performance Evaluation

The Pareto fronts obtained by NSGA-II and MQHDE after 10000 function evaluations are shown in Figure 1. The visual analysis of the Pareto fronts reveals that MQHDE is able to strike a balance between exploration and exploitation of the search space in terms of convergence and diversity. NSGA-II is able to converge better but not succeeded in maintaining the diversity. And also the range of solutions obtained by MQHDE is higher than NSGA-II.

The mean and standard deviation of the all the three metrics for both the algorithms are listed in Table 1. It is apparent from the reported results in Table 1 that the performance of MQHDE is superior to NSGA-II in terms of achieving convergence and in maintaining good diversity in the solutions. The smaller values of the Generational distance and spread of MQHDE indicates the exploitation and exploration capabilities of the algorithm. The larger value of Hypervolume metric measuring both the convergence and diversity also proves the efficacy of the algorithm over NSGA-II. The smaller values of the standard deviation in all the three metrics, proves the consistent performance of MQHDE.

Figures 2, 3 and 4 shows respectively the box plots of Generational Distance, Spread and Hypervolume indicators found by NSGA-II and MQHDE. We can see that the values obtained by MQHDE are superior to the ones obtained by NSGA-II. Furthermore, as the notches in each box plot do not overlap, we can conclude, with 95% confidence, that the true medians do differ.

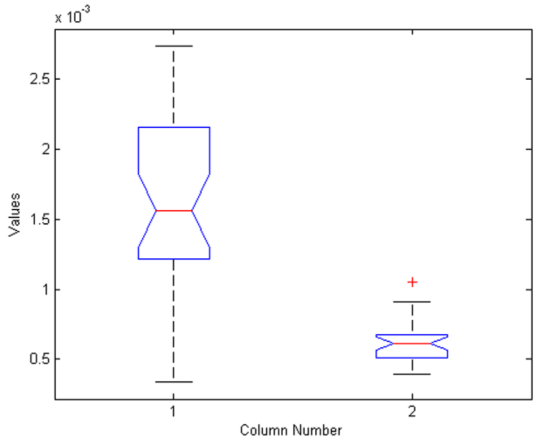


Figure 2 : Box plot of Generational Distance indicator by (1) NSGA-II (2) MQHDE

5. CONCLUSION

This paper presents a new state-of-the-art elitist Multi-objective Quantum-inspired Hybrid Differential Evolution Algorithm for the solution of MONRP which is a vital problem in Search-Based Software Engineering and has broad applicability in software and manufacturing industries. MQHDE combines the strengths of Quantum Computing, Differential Evolution and Genetic Algorithms to maintain the right balance between exploration and exploitation of the search space. The designed mutation operator is adaptive and quite versatile making the approach almost parameter free and eeasy to use. The efficacy of MQHDE for the solution of Multi-Objective Next release problem is evaluated on a MONRP problem pertaining to a real-world banking application. The features of MQHDE help it delivers consistent performance in terms of convergence to the optimal front, maintaining good spread among the Pareto-optimal solutions, and fast convergence compared to NSGA-II. The comparison of MQHDE and NSGA-II is based upon the obtained Pareto fronts, range of extreme points and the performance indicators –Generational Distance (GD), Spread (Δ) and Hypervolume (HV). The results not only prove the effectiveness and efficiency of MQHDE in the solution of

MONRP but also indicate the superior performance of MQHDE over NSGA-II. The performance of MQHDE is promising and therefore can be used with confidence for the solution of even larger real-world instances of MONRP.

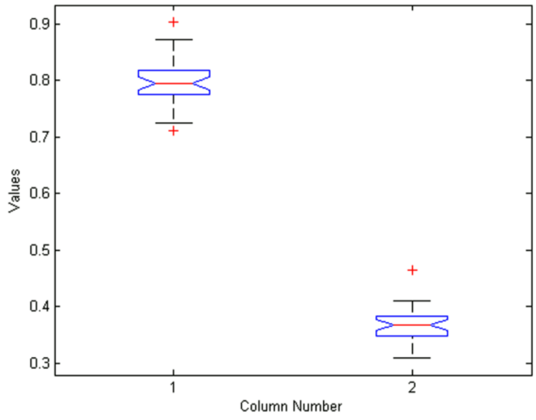


Figure 3 : Box plot of spread indicator by (1) NSGA-II (2) MQHDE

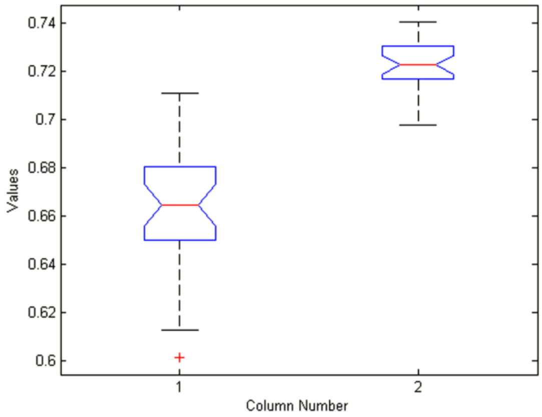


Figure 4 : Box plot of Hypervolume indicator by (1) NSGA-II (2) MQHDE

6. ACKNOWLEDGMENTS

The authors are extremely grateful to Revered Prof. P. S. Satsangi, Chairman, Advisory Committee on Education, Dayalbagh Educational Institutions, for continued guidance and support.

7. REFERENCES

[1]. Mark Harman, Bryan F.Jones. 2001. "Search-based software engineering", Information and software Technology, 833-839.

[2]. A. Bagnall, V. Rayward-Smith, and I.Whittley. 2001. "The next release problem", Information and software technology, 883-890.

[3]. J. Aguilar-Ruiz, I. Ramos, J. C. Riquelme, and M. Toro. 2001. "An evolutionary approach to estimating software development projects", Information and Software Technology, 875–882.

[4]. R. Lutz. 2001. "Evolving good hierarchical decompositions of complex systems", Journal of Systems Architecture, 613–634.

[5]. P. McMinn, M. Harman, D. Binkley, and P. Tonella. 2006. "The species per path approach to search-based test data generation", International Symposium on Software Testing and Analysis, 13–24.

[6].R M Hierons, Z Li, M Harman. 2008. "Search Algorithms for Regression Test Case Prioritization", IEEE Transactions on Software Engineering, 225-237.

[7]. M. O'Keeffe and M. O'Cinneide. 2006. "Search-based software maintenance", Conference on Software Maintenance and Reengineering ,249–260.

[8]. D.Greer and G.Ruhe. 2004. "Software release planning : an evolutionary and iterative approach", Information & Software Technology, 243-253.

[9]. Zhang . Y, M. Harman, and A. S. Mansouri, 2007. "The Multi-Objective Next Release Problem", GECCO: proceedings of the 9th annual conference on genetic and evolutionary computation, 1129–1136.

[10]. Durillo, J., J., Y. Zhang, E. Alba, A. J. Nebro, 2009. "A study of the multi-objective next release problem", SBSE: proceedings of the 2009 1st international symposium on search based software engineering, 49–58.

[11]. Finkelstein A, Harman M, Mansouri SA, Ren J, Zhang Y. 2009. "A search based approach to fairness analysis in requirement assignments to aid negotiation, mediation and decision making", Requirement Eng , 231–245.

[12].M. Harman, J. Krinke, J. Ren, and S. Yoo. 2009. "Search Based Data Sensitivity Analysis Applied to Requirement Engineering", Proceedings of the 11th Annual Conference on Genetic and Evolutionary Computation, 1681–1688.

[13].Yuanyuan Zhang, Enrique Alba, Juan J. Durillo, Sigrid Eldh and Mark Harman. 2010 ."Today/Future importance analysis". Proceedings of the 12th Annual Conference on Genetic and Evolutionary Computation.

[14].A.Charan Kumari, K.Srinivas and M.P.Gupta. 2012. "Software Requirements Selection using Quantum-inspired Elitist Multi-objective Evolutionary Algorithm". Proceedings of the IEEE-International Conference on Advances in Engineering, Science and Management, 782-787.

[15].A. Charan Kumari, K. Srinivas and M. P. Gupta, "Software Requirements Optimization Using Multi-Objective Quantum-Inspired Hybrid Differential Evolution", EVOLVE – A Bridge between Probability, Set Oriented Numerics, and Evolutionary Computation II Advances in Intelligent Systems and Computing, 107-120.

[16]. K. Price and R. Storn, 1995. "Differential Evolution – a simple and efficient adaptive scheme for global optimization over continuous spaces", Technical Report, International Computer Science Institute, Berkley.

[17]. Han, K. H. and J. H. Kim, 2002. "Quantum-inspired Evolutionary Algorithm for a Class of Combinatorial Optimization", IEEE Transactions on Evolutionary Computation, 580-593.

[18]. Deb, K., A. Pratap, S. Agarwal, and T. Meyarivan., 2002. "A Fast and Elitist Multiobjective Genetic Algorithm: NSGA-II", IEEE Transactions on Evolutionary Computation, 182-197.

[19] Su, H. and Yang, Y. 2008. "Quantum-inspired differential evolution for binary optimization", The 4-th international conference on natural computation, 341–346.

[20]. Deb, K.. 2001. "Multi-Objective Optimization using Evolutionary Algorithms". Wiley Chichester, UK.