INDIAN JOURNAL OF SCIENCE AND TECHNOLOGY

RESEARCH ARTICLE

Empirical Evaluation of Tetrad Optimization Methods for Test Case Selection and Prioritization

Shweta Singhal[1], Nishtha Jatana[2]*, Geetika Dhand[3], Shaily Malik[2], Kavita Sheoran[3]

1 Assistant Professor (visiting), Indira Gandhi Delhi Technological University, New Delhi, India
2 Assistant Professor, Maharaja Surajmal Institute of Technology, New Delhi, India
3 Associate Professor, Maharaja Surajmal Institute of Technology, New Delhi, India

OPEN ACCESS

Received: 04-01-2023
Accepted: 12-03-2023
Published: 06-04-2023

Citation: Singhal S, Jatana N, Dhand G, Malik S, Sheoran K (2023) Empirical Evaluation of Tetrad Optimization Methods for Test Case Selection and Prioritization. Indian Journal of Science and Technology 16(14): 1038-1044. https://doi.org/10.17485/IJST/v16i14.2109

* **Corresponding author**.

nishtha.jatana@gmail.com

Funding: None

Competing Interests: None

Published By Indian Society for Education and Environment (iSee)

ISSN
Print: 0974-6846
Electronic: 0974-5645

Abstract

Objectives: Software researchers have been taking advantage of various evolutionary optimization approaches by digitizing them. Test case selection and prioritization based on fault coverage criteria within a time-constrained environment is important in regression testing problem. **Methods**: This work empirically evaluates different approaches that includes evolutionary approaches (Ant Colony Optimization, Bee Colony Optimization, a combination of Genetic Algorithms and Bee Colony optimization), and a Greedy approach. These tetrad techniques have been successfully applied to regression testing. Also, tools have been developed for their implementation. Eight open-source test programs, written in C language have been used for empirical evaluation of the regression testing approaches. **Findings**: The accuracy achieved by t-GSC, being a greedy technique, was found to be least; while that of ACO was found to be the best. All the tetrad approaches yielded borderline better or worse results, while all the four gave excellent time and size gains. **Novelty**: There are many studies available in the literature that compare various regression testing approaches of a similar kind. Instead of repeating the same, it is intended to evaluate two well-accepted approximation approaches: a hybrid approach, and a greedy approach. It has been tried to evaluate the efficiency of the greedy approach with the metaheuristic approach. It is imperative to compare approaches following different algorithmic paradigms, yet trying to solve the same problem.

Keywords: Ant Colony Optimization; Bee Colony Optimization; Genetic Algorithms; Greedy Set Cover; Software Testing; empirical comparison

1 Introduction

Test case selection prioritization performed within a time-constrained environment happens to be a combinatorial optimization problem. Evolutionary techniques like Ant Colony Optimization (ACO)[(1)], Bee Colony Optimization (BCO)[(2)], and Genetic Algorithms (GA)[(3)], etc. are approaches built around the physical behavior of ants,

bees, and human DNA and based on these the test case selection & prioritization has tried to be solved. Along with these a time-constrained greedy set cover (t-GSC) technique has also been taken advantage of in resolving regression test case selection and prioritization [4]. The tetrad evolutionary approaches (ACO, BCO, BCO_GA, and t-GSC) have been empirically compared with each other for 8 sample C and Java programs for the selected and prioritized resultant test suites obtained. Each technique was executed for each of the programs 10 times and the results have then been averaged. The current study tries to compare the four techniques in the area of regression test selection and prioritization.

The problem of TCSP has been researched since the 1990s and it continues. The current paper focuses on time-bound test case prioritization via coverage of faults. Four evolutionary combinatorial optimization approaches have been selected for comparison. Ant Colony Optimization (ACO) was primarily used for TCSP and was implemented in [1]. Similarly, Bee Colony Optimization (BCO) was established for TCSP in [2]. A hybrid approach combining Genetic Algorithms (GA) and BCO was also implemented for TCSP [3]. Recently, in 2019 a greedy approach based on Set Cover in a time-constrained environment was proposed [4]. These four approaches form the basis of this research work. The motivation behind the comparison of only these tetrad approaches is to provide a comparison that has not been done before. ACO and BCO are successfully verified and validated approaches in TCSP, GA with BCO provides a hybrid approach for solving TCSP, whereas the Set Cover approach is a greedy strategy trying to solve an NP-complete problem. Thus, the experiments conducted in this study aim to find out which approaches may provide quick and valid TCSP test suits. The already developed tools for the four approaches were tested on 8 open source programs and the results have been then aggregated.

Natural ants are tiny creatures unable to see, even then they have one of the best exploration capabilities not only in finding the food source but also the shortest path back to their nests. God's creation of their antennas and pheromones helps them accomplish this behavior. Ants use stigmergy to communicate and the rest of the colony converges to the shortest path. ACO is a digital adaptation of the same ant behavior. Artificial or digital ants must possess the ability to synchronize in exploring local solutions based on some previously gathered information and share it among their community so that a global solution can be reached. Figure 1 depicts the technique used by ACO. The concept of ACO was implemented to form a tool ACO_TCSP [1]. ACO is still being used predominantly in the area of TCSP [5].

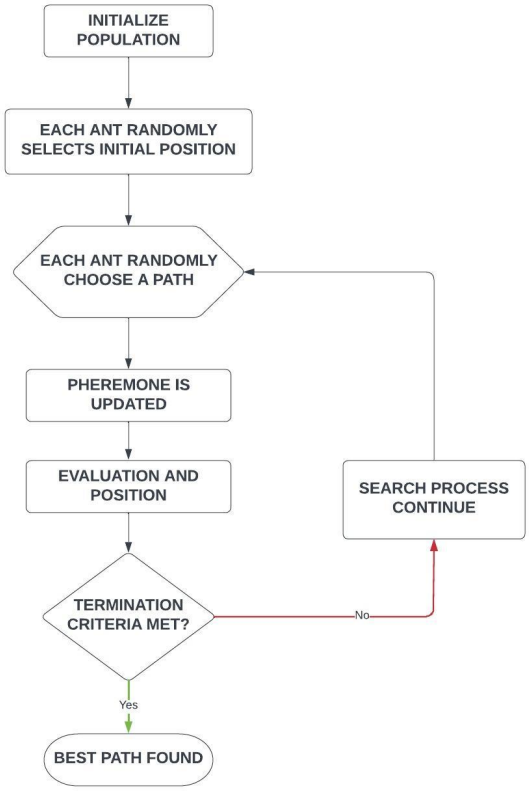

Fig 1. Flowchart of ACO

BCO is a digitized form of the way a colony of honey bees forages for its food. It is a population-grounded search technique having bees as the key employed agents. Different sorts of bees form honey bee comb: Single Queen Bee - Whose role lies in laying eggs to support constructing the hive inhabitants, Male Drone Bees – Which hold the responsibility of breeding alongside the queen bee to assist the hive growth, and Worker Bees - The chief working agents of the bee colony. Hive conservation, bringing food for the colony, and guiding associate bees to reach a source of food, stance as some of the key roles of worker bees. Worker bees are categorized as Scouts and Foragers. Scouts are supposed to explore new food sources while foragers exploit the already explored paths to reach the food source[6]. To search for available food sources, the Scout bees begin randomly from the hive. When they are exhausted, they return hive after their random exploration. The scout bees then share the gathered information with the forager bees via Waggle Dance. The waggle dance is performed in a particular manner to convey information about food source quality, distance, etc. After learning about the explored information from the scout bees, it is now the turn of forager bees to exploit the best food sources and bring back food from the best quality sources. The process of BCO is depicted in Figure 2.

Artificial Bee Colony (ABC) is a well-known technique that is applied to regression TCSP[6]. The BCO algorithm for solving fault-based regression test prioritization within a time-constrained environment was proposed and implemented in [2] respectively. The same implementation forms the basis of BCO in the current research for comparison.

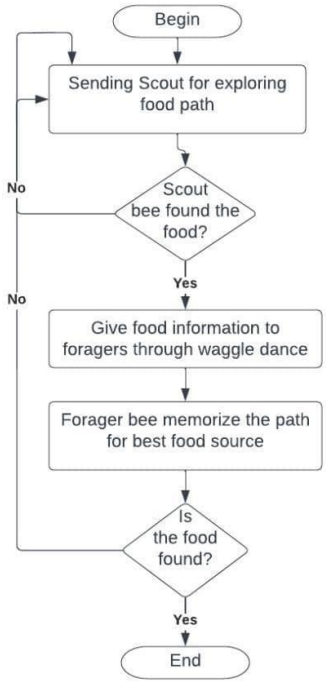

Fig 2. Flowchart of Artificial Bee Colony

Humans keep evolving and adapting to survive. Genetic Algorithms (GA) are an evolutionary approach grounded on survival of the fittest solutions for creating new populations that are fitting and better than the earlier ones. GA was introduced by John Holland[7] . The digital genetic procedure instigates by encoding an initial set of populations into a string format. Fitness equations based on some criteria that solve the problem under consideration are next used on a random population. In case fitness equations can achieve the desired fitness values, the genetic procedure stops, else a novel set of populations is produced via crossover & mutation procedures. This entire process is looped till a pre-decided count of generations is reached or a solution meeting the desired fitness criterion is available.

The power of both techniques (GA and BCO) was combined in a single algorithm to unravel desired solutions for regression test prioritization. This hybrid technique was developed by Suri et al. [3]. The algorithm picks a new set of test cases by examining the prevailing set of initial test cases built on the maximum fault coverage and minimum execution time criterion. The approach was implemented as the MHBG_TCS tool and empirically validated on 17 sample programs. The same algorithm forms the basis for the third approach being compared in the current paper.

The set cover problem has been solved using a greedy approach as early as 1979[8]. The Set Cover problem attempts to decrease the number of test cases via a postponed greedy technique so that a new smaller set of test cases realizing each of the necessities achieved with the initial test cases can be discovered[9]. The initial proposal for solving TCSP using a set cover with the help of mutation testing was proposed in[10]. The technique was then improved by running it in a time-constrained environment and adding some selection criteria. The running interval of each test case was supplemented to narrow down the process of TCSP in addition to the current criteria of killing maximum faults with the least possible number of test cases chosen to produce the optimized test suite. A new technique, named 't-GSC' was thus established to form the improved test suite[4], this same approach has been used for comparison in the current study.

Over the three decades of research, many techniques and algorithms have been developed and tested for TCSP[11–14]. Most of the studies focus on suggesting and implementing new techniques, or survey the existing similar types of approaches. Our aim is to evaluate the different kinds of approaches to study the nature of their performance.

2 Methodology

Advancements in the area of Regression TCSP can be tracked from a recent study[12]. Quantum-based PSO has recently been employed for Regression TCSP[13]. Ahmed et al. investigated the literature to gather studies related to value-based cost-cognizant TCP techniques to investigate the research gaps in the area. The ACO_TCSP tool along with the BCO, MHBG_TCS, and t-GSC tools have already been developed and all four techniques take the faults killed and the running time of the test cases as their input. The output is an optimized test suite. All the tools yield a test suite, yet the answers are slightly varying for all the techniques. The tools have been run on eight open-source programs (details are given in the following section). Mutation faults were induced in all the programs and test cases generated. Then the fault matrix for each program was furbished to create the input required for the tools under comparison. Since three of the four techniques being compared are approximation techniques, they were run 10 times for every program and the results were then averaged for analysis. For analysis, eight different open-source C++ and JAVA programs have been considered. Ephemeral details about the Programs Under Test (PUT) like their length, and versions (number of faults) are summarized in Table 1.

Table 1. Details of Benchmark Programs used

PUT No.	PUT Name	Size	Versions used	Total Test Cases
PUT1	College_Admission	281	Five	9
PUT2	Hotel_Management	666	Five	5
PUT3	Triangle_sides	37	Six	19
PUT4	Quadratic_eqn	38	Eight	19
PUT5	Publishing_cost	31	Eight	19
PUT6	Calculator_	101	Nine	25
PUT7	Previous_day	87	Seven	19
PUT8	Railway_booking	129	ten	26

3 Results and Discussion

To establish our confidence in the soundness of the techniques under comparison, three aspects were compared. The amount of size efficiency, time gain, and percentage accuracy achieved by each of the four techniques were recorded and plotted. All four techniques give highly motivating results with each of the techniques being superior to the other in one or the other aspect. These have been analyzed and the reasons explaining this behavior have been tried to be looked upon. Such results for the individual techniques may be found in separate studies, but a combined comparison is missing in the literature.

The amount of test cases forming the final test suite is lesser in comparison to the overall amount of test cases building the original test suite. By what amount has the number of test cases selected been reduced is tried to be analyzed using the Averaged Percentage Size Efficiency (APSE) calculated as follows:

$$\text{APSE} = \left(\frac{|TS| - |RTS|}{|TS|} \right)^{*} 100 \tag{1}$$

Where TS represents the Total Size of the initial Test Suite, RTS is the size of the Resultant Test Suite. APSE has been calculated for all ACO, BCO, hybrid BCO_GA, and t-GSC tools and the details have been plotted in Figure 3. As can be observed from

the line graph that all 4 tools provide almost similar size efficiency varying from 60% to 85%. The only exception is the greedy t-GSC approach which is found to underperform as compared to the approximation approaches. BCO_GA gave the best APSE for all 8 programs. 79.8% APSE was achieved by BCO_GA, ACO and BCO for P1, while only 60% APSE could be achieved by t-GSC for the same program. For P2, P5 and P7, all the four approaches gave 60%, 90% and 85% APSE respectively, which is also the optimal possible APSE that could be achieved considering complete fault coverage for the given test suites. ACO and BCO_GA yielded best 90%, 85%, and 82.6% APSE for P3, P4 and P8 respectively, while BCO gave slightly less APSE of 82.5%, 76% and 78% for the same programs. Except for P6, for which ACO resulted in 62% APSE as against 75% achieved by BCO_GA, both ACO and BCO_GA gave best results.

Fig 3. APSE (Average Percentage Size Efficiency)

Averaged Percentage Time Gain (APTG) has been calculated using the formula (2) and the aggregated results presented in Figure 4.

$$\text{APTG} == \left(\frac{RT - RRT\ |}{ET}\right) * 100 \tag{2}$$

Where RT is the Total Running Time for the entire test suite of the PUTs (P1 – P8), RRT is the average Running Time for the Resultant test suite. Yet again the plot of Figure 4 establishes the similarity in time gain achieved using the four approaches. P1, P2, P5 and P7 got APTG values of approximately 80%, 58%, 89% and 82% respectively for all the four approaches. For P3, P4, P6 and P8 ACO outperformed BCO and BCO_GA approaches although by <10% variation. ACO yielded APTG of 89.8%, 81%, 72% and 82% for P3, P4, P6 and P8 respectively. Hence it can be observed that except for minor variations, all of the ACO, BCO, hybrid BCO_GA, and the t-GSC provide highly inspiring time gains as against running the entire test suite.

Fig 4. APTG (Averaged Percentage Time Gain)

Averaged Percentage Accuracy _(APA) has been designed using the formula (3) and the gathered results are presented in Figure 5.

$$\text{APA} = \left(\frac{OR}{10}\right) * 100 \tag{3}$$

Where, OR is the average number of Optimal Runs of the respective tool.10 is the total number of runs the tool has executed for each program (P1-P8). Optimal run refers to the output which is possibly the best out of all. Since the eight programs under test are small, thus optimal output could be known and compared. Figure 5 represents the radar plot for APA of the four approaches. The desired is a complete hexagonal plot with a maximum radius that would represent 100% accuracy. But since ACO, BCO and hybrid BCO_GA are approximation approaches; thus 100% accuracy is not possible. Hybrid BCO_GA gave very good accuracy while ACO gave the best accuracy out of all tetrad techniques. Due to the very greedy nature of the t-GSC technique, it gave the worst accuracy for regression TCSP. This is expected because TCSP is an NP-Complete problem.

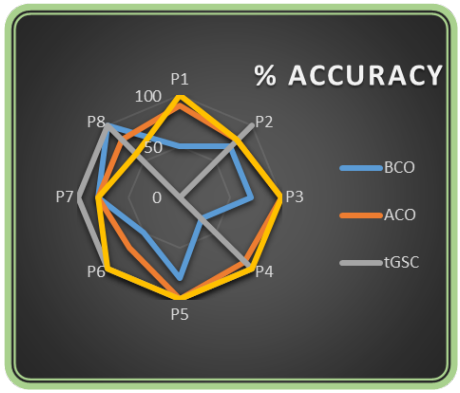

Fig 5. APA-Averaged Percentage Accuracy

The current study made comparison among a greedy approach (t-GSC), a hybrid approach (BCO-GA) and two well established nature inspired optimization approaches (ACO and BCO). Such a comparison has not been found till date. The tetrad approaches: t-GSC, hybrid BCO_GA, BCO, and ACO were tested successfully using eight test programs for regression TCSP. All four tools were run 10 times on every test program and the averaged results have been investigated. Our experimental research led to the conclusion that the hybrid BCO_GA approach was found to be better than the individual ACO, BCO, and greedy set cover approaches in terms of size efficiency. BCO_GA achieved a hopping average APSE of 80.625% for the 8 programs. The greedy set cover approach tends to provide very quick results, but the results were far from optimal, hence the authors would not recommend the use of the t-GSC approach alone for TCSP. Nevertheless, it may/may not provide good results when combined with an existing search-based approach. Also, it can be said that instead of using an individual evolutionary approach, hybrid approaches may provide better solutions to optimization problems. Future investigations can be made on more evolutionary algorithms with larger data sets.

References

1) Suri B, Singhal S. Implementing Ant Colony Optimization for Test Case Selection and Prioritization. *International Journal of Computer Science and Engineering*. 2011;3:1924–1932. Available from: https://www.academia.edu/28114634/Implementing_Ant_Colony_Optimization_for_Test_Case_Selection_and_Prioritization.
2) Kaur A, Goyal S. Implementation and Analysis of the Bee Colony Optimization algorithm for Fault based Regression Test Suite Prioritization. *International Journal of Computer Applications*;41:1–9. Available from: https://doi.org/10.5120/5606-7867.
3) Suri B, Singhal S. Evolved regression test suite selection using BCO and GA and empirical comparison with ACO. *CSI Transactions on ICT*. 2016;3. Available from: https://link.springer.com/article/10.1007/s40012-016-0080-5.
4) Jatana N, Suri B. An Empirical Comparison of t-GSC and ACO_TCSP Applied to Time Bound Test Selection. *Recent Advances in Computer Science and Communications*. 2021;14(2):555–563. Available from: https://doi.org/10.2174/2213275912666190417152016.
5) Lu C, Zhong J, Xue Y, Feng L, Zhang J. Ant Colony System With Sorting-Based Local Search for Coverage-Based Test Case Prioritization. *IEEE Transactions on Reliability*. 2020;69(3):1004–1020. Available from: https://doi.org/10.1109/TR.2019.2930358.
6) Nayak S, Kumar C, Tripathi S, Mohanty N, Baral V. Regression test optimization and prioritization using Honey Bee optimization algorithm with fuzzy rule base. *Soft Computing*. 2021;25(15):9925–9942. Available from: https://doi.org/10.1007/s00500-020-05428-z.
7) Holland J. Genetic Algorithms and Adaptation. In: Adaptive Control of Ill-Defined Systems;vol. 16. Springer US. 1984;p. 317–333. Available from: https://doi.org/10.1007/978-1-4684-8941-5_21.
8) Chvatal. A Greedy Heuristic for the Set-Covering Problem. *Mathematics of Operations Research*. 1979;4(3):233–235. Available from: https://doi.org/10.1287/moor.4.3.233.
9) Tallam S, Gupta N. A concept analysis inspired greedy algorithm for test suite minimization. *ACM SIGSOFT Software Engineering Notes*. 2006;31(1):35–42. Available from: https://doi.org/10.1145/1108768.1108802.
10) Jatana N, Suri B, Kumar P, Wadhwa B. Test Suite Reduction by Mutation Testing Mapped to Set Cover Problem. In: Proceedings of the Second International Conference on Information and Communication Technology for Competitive Strategies. ACM. 2016. Available from: https://doi.org/10.1145/2905055.

2905094.
11) Singhal S, Suri B. Multi objective test case selection and prioritization using African buffalo optimization. *Journal of Information and Optimization Sciences*. 2020;41(7):1705–1713. Available from: https://doi.org/10.1080/02522667.2020.1799514.
12) Jatana N, Suri B. Application of Nature Inspired Algorithms to Test Data Generation/Selection/Minimization using Mutation Testing. *Artificial Intelligence and Natural Algorithms*. 2022;p. 213–249. Available from: https://doi.org/10.2174/9789815036091122010016.
13) Jatana N, Suri B. An Improved Crow Search Algorithm for Test Data Generation Using Search-Based Mutation Testing. *Neural Processing Letters*. 2020;52(1):767–784. Available from: https://doi.org/10.1007/s11063-020-10288-7.
14) Habib AS, Khan SUR, Felix EA. A systematic review on search-based test suite reduction: State-of-the-art, taxonomy, and future directions. *IET Software*. 2023. Available from: https://ietresearch.onlinelibrary.wiley.com/doi/pdfdirect/10.1049/sfw2.12104.