Journal of Information Systems Engineering and Management
2025, 10(13s)
e-ISSN: 2468-4376
https://www.jisem-journal.com/

Research Article

A Hybrid Multi-Objective Evolutionary Optimization Algorithm for Next-Release Problem

Divya K V1, R J Anandhi2

1Research Scholar, New horizon College of Engineering, Bengaluru, Visvesvaraya Technological University (VTU), Belagavi – 590018

Email: divya.k.vasudevan@gmail.com

2Research Supervisor, New horizon College of Engineering, Bengaluru, Visvesvaraya Technological University (VTU), Belagavi – 590018. Email: rjanandhi@hotmail.com

ARTICLE INFO

Received: 28 Nov 2024

Revised: 09 Jan 2024

Accepted: 31 Jan 2024

ABSTRACT

The software requirements selection process, a crucial step in software development, is a complex task. It involves identifying the most beneficial set of requirements for a software release while adhering to budget constraints. This problem, known as the next release problem (NRP), is challenging and classified as a non-deterministic polynomial (NP) hard problem. Interdependencies and other limitations further complicate the specified criteria. Selecting a specific set of requirements for the upcoming software release is a computationally challenging issue, falling under the category of NP-Hard problems. This paper proposes a hybrid method called MO-ACO-DE, which combines Multi-Objective Ant Colony Optimization with Differential Evolution to solve the multi-objective NRP. This work defines the NRP as a multi-objective optimization problem with two challenging objectives: customer satisfaction and development cost. Additionally, three constraints are introduced to address two real-world instances of the NRP. The proposed approach combines the management techniques of ant colony optimization (ACO) with the operators of the differential evolution (DE) algorithm to balance the exploitation and exploration stages of the optimization process. Both benchmark and real-world classic and realistic datasets were used for the experimental analysis of the proposed model. The results indicate that MO-ACO-DE outperforms other methods and enhances the fairness of requirement selection, especially when budget limitations are decreased.

Keywords: software requirements, multi-objective, next release problem, NP-hard, ant colony optimization, differential evolution.

INTRODUCTION

Requirement elicitation and determination are the initial stages in software engineering that are crucial for software development. Requirement engineering is the systematic procedure of collecting, examining, and recording client software requirements. A sequential elicitation process produces a list of software development needs. Getting accurate search results that meet client needs is a major obstacle for requirement engineers. This problem arises from inadequate communication, leading to poor satisfaction rates and increased project costs. The incremental development technique is a highly renowned strategy for software development in the field of software engineering, and it is particularly well-suited for projects of a large scale. This process is employed to create the program in several iterations. Each release corresponds to a certain set of requirements exclusively established by the software firm. Developing all of the provided needs is challenging due to limitations in the project budget, project delivery timeline, technology, and inherent requirement conflicts [2]. Choosing an appropriate subset of criteria is exceedingly challenging and prone to mistakes in large-scale projects. Hence, it is important to have a technique to determine the suitable subset of requirements that may aid the development team in making more informed choices [3]. Numerous applications have extensively utilized artificial intelligence and stochastic learning [4-10]. Optimization plays a significant role, directly or indirectly, in several applications of artificial intelligence and stochastic learning [11–18]. Challenges with software could be seen as exemplary instances of these applications. It is well-recognized that in today's corporate environment, consumer relationships and requirements are subject to ongoing change and increasing complexity [19]. To thrive in today's highly competitive marketplace, software businesses must regularly update their current software versions. Selecting the appropriate needs for inclusion in the next software version is a crucial aspect of the requirements analysis process for a software project. This ensures that the firm maximizes its profits while controlling costs [20]. Sometimes, a client may make a request that contradicts another client's request. Needs are classified based on the necessary characteristics

Copyright © 2024 by Author/s and Licensed by JISEM. This is an open access article distributed under the Creative Commons Attribution License which permitsunrestricted use, distribution, and reproduction in any medium, provided the original work is properly cited.

and relationships [21, 22]. The interdependencies between requirements, the firm budget, and the constrained timeline make it impossible to satisfy all client demands. What are the prerequisites for the next software release? The Next Release Problem (NRP) is a single objective issue that was initially posed by Bagnall in 2001 and is NP-Hard (NPH) [23]. As a result, meta-heuristic algorithms are often used to tackle these problems. Heuristic search employs stochastic elements to discover optimal solutions for optimization problems without exhaustively exploring the full search space [24]. As mentioned in [25], population-based algorithms are meta-heuristic algorithms that work with a set of solutions throughout each iteration. Nature-inspired optimization algorithms are a substantial subset of population-based algorithms. Their motivation stems from natural evolution, and their objective is to manage a set of solutions in each generation of the optimization process [26]. Nature-inspired optimization algorithms, such as evolutionary, physics, and swarm-based algorithms, can tackle complex problems in several fields [27].

Multi-objective optimization is a process that simultaneously optimizes two or more objective functions. The multi-objective optimization problem results in a collection of answers known as the Pareto optimal solutions set [5]. Search-based software engineering has experienced significant expansion in recent years. This approach uses search-based algorithms to address a range of difficulties. This research presents a method for addressing the Multi-Objective Network Routing Problem (MNRP) using a combination of a multi-objective ant colony optimization and a differential evolution algorithm (MO-ACO-DE). The MO-ACO-DE method integrates the features of ant colony optimization (ACO) and differential evolution (DE) algorithms. The hybridization method facilitates the optimization process's efficient use, exploration, and convergence. Furthermore, this novel method improves multi-ACO performance by efficiently balancing global and local search capabilities. The aim is to showcase that the suggested methodology successfully acquires optimal solutions for MO-NRP while preserving the variety of choices on the Pareto front. To accomplish efficient requirement selection, MO-ACO-DE integrates many objectives and constraints. The MO-ACO-DE method achieved superior results to DE, ACO, and other state-of-the-art algorithms. The research's main contributions are succinctly outlined below:

- The model effectively manages intricate requirement dependencies in NRP, leveraging ACO's combinatorial optimization strengths and DE's local search enhancements.
- Combining ACO with DE leads to high-quality, diverse solutions, providing decision-makers with a wide range of optimal trade-offs.
- The hybrid model speeds up convergence towards the Pareto front using DE's mutation and crossover, enhanced by ACO's guided search process.
- The hybrid approach is scalable and adaptable, suitable for various NRP instances, from small projects to large software systems, ensuring broad applicability and practical utility in release planning.

The subsequent sections of the article are structured as follows: Section 2 offers an overview of previous research and studies in the field. Section 3 delineates the Multi-Objective Next Release Problem (MO-NRP) and details the formulation of the multi-objective NRP. Section 4 introduces the proposed MO-ACO-DE algorithm tailored for MONRP. Section 5 discusses the dataset and evaluation measures and presents experimental results. Section 6 concludes the findings and insights derived from the study.

LITERATURE REVIEW

The next release issue was presented in SBSE as an optimization difficulty [28]. Conventional techniques including simulated annealing (SA), greedy algorithms, linear programming, and hill-climbing (HC) have been used to solve the NRP [28,29]. These approaches consider the NRP as a single-objective problem by combining the goals. The Multi-Objective Next Release Problem (MNRP) was subsequently defined using the Multi-Objective Programming (MOP) framework [30]. Various approaches to solving the MNRP problem have been investigated, including the use of multi-objective Genetic Algorithms (GA), Non-dominated Sorting Genetic Algorithm II (NSGA-II), Pareto Envelope-based Selection Algorithm (PESA), and Strength Pareto Evolutionary Algorithm II (SPEA2) [7,8]. Furthermore, the Whale Optimisation Algorithm (WOA), Grey Wolf Optimiser (GWO), NSGA-II, and SPEA2 [31,32] have been recently utilized in various studies. Genetic algorithms have effectively solved the NRP [33] and MNRP [34]. Evolutionary methods such as Teaching-Learning-Based Optimisation (TLBO), multi-objective Artificial Bee Colony (MABC), and Ant Colony Optimisation (ACO) have been suggested to address certain cases of MONRP [35]. Chavez et al. [36] presented a bi-objective Teaching-Learning-Based Optimisation (TLBO) algorithm and created a multi-objective Differential Evolution (DE) approach utilizing Pareto tournaments [37].

In [38], the authors emphasize the applicability of NRP within a constrained goal model, highlighting the hierarchical nature of needs and their interdependencies. Their study focused on achieving specific goals using inference models to explore POSS. In another study, [39] the MONRP was addressed using an enhanced binary Particle Swarm Optimization (PSO) algorithm combined with a greedy-random swarm initiation technique. This enhancement incorporated priority relations among criteria and introduced additional

velocity vectors per particle. Their fitness function weighted overall satisfaction and development costs of chosen needs, aiming for specific optimal solutions rather than POSS. However, MOPSO algorithms, known for rapid convergence, face challenges of premature termination and false Pareto optimal solutions in multi-objective optimization [40]. Researchers in [41] have presented a method by which a virtual savant (VS) can independently acquire the capacity to solve the fundamental NRP accurately and efficiently. This method encompasses a general problem-solving approach that integrates machine learning and heuristics to emulate the solution-generation process employed by a reference program for various problem instances.

In [42], researchers proposed a hybrid approach that combines a multi-objective optimization strategy with the Monte Carlo simulation method. This method aims to find solutions that optimize multiple objectives while considering the uncertain and stochastic nature of the problem through simulation. In another study [43], an architecture for NRP was introduced. This approach incorporates interactive optimization and machine learning techniques to improve planning. Specifically, the method utilizes an interactive genetic algorithm, enabling decision-makers to interact with the optimization process. It leverages professional expertise to guide both the learning and decision-making processes effectively. The equation was modified by authors in [44] to include error detection and the learning curve. The optimum release problem is addressed in [45] using Bayesian approaches and expert insights. In [46], decision quality is assessed by analyzing the ratio between the cost of a release choice and the actual optimal solution, which can only be determined when all data is considered. Furthermore, they analyzed a more cautious approach that required the release recommendation to stay relevant throughout many observation periods. Kumar et al. [47] introduced a reliability growth model that uses software patching to improve the dependability and cost-effectiveness of the software system. An innovative method was introduced in [48] to assess the reliability of multi-release open-source software (OSS) by using generic masked data. Their methodology differs from standard methods by employing an additive model, which can efficiently address various instances of obscured data. A methodology was proposed in [49] that utilizes both the real and anticipated release time to optimize the expense of timing the release during execution.

Several software aspects possess qualitative characteristics that are challenging to measure. Estimating the program's cost is exceedingly challenging, even with rough approximation. Harman et al. devised a precise strategy known as OATSAC1 for conducting sensitivity analysis using rigorous procedures[50]. In 2019, Domínguez-Ríos et al. presented five novel precise algorithms. Unlike prior algorithms that did not consider spread solutions, they discovered a collection of well-distributed solutions during the search process [51]. In addition, precise techniques have been employed to address the challenge of selecting the most suitable requirement for a particular aim [29].

[31] conducted a study using a mix of multi-objective grey wolf and whale optimization techniques, along with three additional evolutionary optimization algorithms, to tackle the MNRP problem. Their research includes a comparative analysis of Pareto fronts derived from eight distinct quality matrices.

Consequently, their findings demonstrated that MOWOA outperforms other methods. A recent work introduced a new meta-heuristic method dubbed the binary artificial algae algorithm [52]. This approach was designed to pick the best subset of criteria. Their findings from two imprecise datasets illustrate that the proposed method generates sets without human errors. The Nautilus is a freely available tool introduced in [53] to tackle selecting needs that will be implemented in the upcoming release stage. This is utilized to showcase the main features of Nautilus and how they may be customized to address a software engineering challenge.

In [54], a model using the Bezier Curve and multimodal delayed PSO [55] was proposed to create smooth pathways in a discrete environment instead of linking the center of the grids with linear segments. By employing this technique, the mobile robot may effectively track a curved trajectory instead of a segmented straight route, which is advantageous for several applications. The A∗ approach was utilized by authors in [56] to determine the best grids for creating a segmented linear route. Afterward, a least square policy iteration was used to optimize the placement of the selected grids within a defined radius and produce a smooth trajectory. Although the RRT is less effective, the hybrid approach produces much better outcomes. Nevertheless, the least square optimization is limited to a specified radius encompassing the pre-selected places. In addition, the authors neglected to provide a comparison study between A∗ and their suggested algorithm to demonstrate its effectiveness.

The NRP has evolved from being treated as a single-objective problem with methods like simulated annealing, greedy algorithms, linear programming, and hill-climbing to being addressed as a MONRP. Various evolutionary algorithms have been explored for MONRP, including multi-objective Genetic Algorithms (GA), NSGA-II, PESA, SPEA2, Whale Optimization Algorithm (WOA), Grey Wolf Optimizer (GWO), Teaching-Learning-Based Optimization (TLBO), and multi-objective Artificial Bee Colony (MABC). Recent advancements include hybrid approaches combining optimization strategies with Monte Carlo simulation, machine learning techniques, and interactive genetic algorithms to enhance decision-making. The studies

highlight the importance of balancing exploration and exploitation, handling uncertainties, and integrating expert knowledge to improve solution quality and dependability. Various models and algorithms have demonstrated the ability to handle the complex, hierarchical, and interdependent nature of NRP, yet challenges like premature convergence, false Pareto solutions, and computational efficiency remain.

A hybrid multi-objective evolutionary optimization algorithm for NRP offers several advantages. Combining multiple evolutionary algorithms can balance exploration and exploitation more effectively, leading to higher-quality and more diverse solutions. Integrating different optimization strategies, such as DE and ACO, can accelerate convergence and reduce computational time, making the algorithm suitable for large-scale problems. The hybrid algorithm can better handle the stochastic nature and uncertainties inherent in NRP by incorporating techniques like Monte Carlo simulation and machine learning. Additionally, leveraging interactive genetic algorithms allows decision-makers to dynamically guide the optimization process, incorporating their expertise and preferences to refine solutions continuously. The hybrid approach can be adapted to various problems, from small projects to complex software systems, ensuring broad applicability and practical utility in real-world scenarios. By combining multiple optimization techniques, the hybrid model can effectively manage and optimize the intricate dependencies between requirements, enhancing the robustness and adaptability of the solutions.

Preliminary Concepts

Multi-Objective next release problem

In practical situations, optimization problems include determining the values of decision variables that can achieve one or more objectives by minimizing or maximizing them. Single-objective problems can have a singular target or utilize weighted aggregation techniques to merge several objectives into a central objective [57]. When an issue has more than two objectives, the fundamental goal of the optimization problem is to combine these objectives and find a single solution. Conversely, challenges related to multi-objective optimization have a diverse range of solutions. Nevertheless, a set of options known as Pareto front solutions are regarded as non-dominated solutions (NDS). These systems may optimize many goal functions concurrently, even with competing objectives. Mathematically, multi-objective optimization problems can be represented by a vector encompassing constraints, objective functions, and decision variables. Decision-makers aim to either minimize or maximize the goal functions. The multi-objective optimization problem seeks to determine the vectors that provide the optimal values for all objective functions while still adhering to the constraints of the job. For example, this issue has □ objectives and two solutions: □ = □$_1$, □$_2$,.. □$_□$and □ = □$_1$, □$_2$,.. □$_□$. □is considered to dominate □ and be part of the Pareto front only if □ is superior to or equal to □ in all goals *1, 2,* □, and □ is certainly superior to □ in at least one objective. Alternatively, neither of the two solutions has a clear advantage; the Pareto front is the collection of alternatives that are not dominated by any other solution.

Formulation of Multi-Objective NRP

Consider a group of consumers □ = {□$_1$, □$_2$, … , □$_□$}for a software organization responsible for addressing the requirements □ = {□$_1$, □$_2$, … , □$_□$}. The software system assumes that all the requirements are independent to avoid conflicts. It is necessary to allocate certain requirements to meet the consumer requests. The defined requirement's associated cost can be computed by translating the resources required to accomplish it into cost terms □ = {□$_1$, □$_2$, … , □$_□$} here, b denotes the budget. Each consumer is associated with a weight factor representing the degree of significance □ = {□$_1$, □$_2$, … , □$_□$}, the weights are between 0 to 1 □$_□$ =∈ [*0,1*], and the summation of weights is 1 $\sum_{□=1}^{□}$ □$_□$ =∈ [*0,1*]. Based on all the factors and objectives, the overall score for each consumer to satisfy the requirement is computed as the dot product between weights and consumer value □(□$_□$, □$_□$) shown in the below equation

$$□_□ = \sum_{□=1}^{□} □_□ \cdot □(□_□, □_□)$$

Each client has a weight factor that may be used to show how important they are to the business and based on the decision vector $\overrightarrow{□}$ the requirement is released, 0 represents no release and 1 represents release.

MO-NRP refers to modifying the regular NRP to incorporate additional objectives while working within a limited budget. Existing research has covered several optimization approaches offered for two to three MO-NRP. For instance, let's examine the Multi-Objective Nonlinear Resource Planning (MO-NRP) problem, which has two objectives: maximizing customer profit and minimizing cost. The MO-NRP follows the same technique as NRP, with the distinction that it optimizes two or multiple objective functions using the score function, as seen below.

$$\square\square_{\square} = \sum_{\square=1}^{\square} \square_{\square} \cdot \square_{\square}$$

$$\square\square_{\square} = \sum_{\square=1}^{\square} \square_{\square} \cdot \square_{\square}$$

In the above equation $\square\square_{\square}$ and $\square\square_{\square}$ represents two objectives, profit, and cost, respectively, $\square_{\square}$and $\square_{\square}$ are score and cost. To enhance the selection of needs in the MO-NRP scenario, it is important to consider many objective aspects, such as maximizing consumer benefit, minimizing cost, ensuring equality, and selecting requirements appropriately. As mentioned before, the MO-NRP has only been considered for two purposes. In [58], a multi-objective optimization approach has been proposed, considering more than two objective parameters by enlarging the search area. The mathematical definition of MO optimization is as follows: $\square$ represents a generic variable, and $\overrightarrow{\square}$represents the decision vector.

$$\overrightarrow{\square}(\square) = [\square_1(\square), \square_2(\square), \ldots \ldots \square\square(\square)]\square$$

$$\square\square(\square) = 0; 1 \leq \square \leq \square$$

$$\square\square(\square) \geq 0; 1 \leq \square \leq \square$$

In the given equation, $\square$ denotes the number of objectives to be taken into account during the design of the NRP, while $\square$ and $\square$ represent the equality and inequality constraints, respectively.

PROPOSED METHODOLOGY

Ant Colony Optimization (ACO)

ACO is a sophisticated and flexible metaheuristic algorithm. This approach draws inspiration from the foraging habit of ants. It has proven particularly effective in solving challenging optimization problems, such as the Next Release Problem (NRP) in software engineering. The NRP involves meticulously selecting high-quality software features to be included in the next release to maximize customer satisfaction while considering constraints such as cost and resources. The issue may be precisely shown as a graph, where nodes represent attributes and edges indicate possible transitions between these attributes.

Multi-Objective Ant Colony Optimization (MO-ACO)

Multi-Objective Ant Colony Optimisation (MO-ACO) is a sophisticated method designed to address the complexities of choosing software features for the next release. It can optimize many competing objectives simultaneously in the Next Release Problem (NRP). The NRP is a prevalent issue in software engineering. It involves effectively managing several factors, such as optimizing customer happiness, minimizing expenses, and conforming to limitations on available resources. MO-ACO utilizes the behavior of ant colonies to navigate a complicated decision-making environment. It achieves this by utilizing pheromone trails and heuristic information to identify a collection of Pareto-optimal solutions.

Problem Definition

Within the framework of NRP, the multi-objective optimization issue entails choosing a subset of features $\square \subseteq \square$ from the pool of accessible features $\square$. The objectives generally consist of optimizing customer satisfaction $\square_1(S)$, minimizing development cost $\square_2(S)$ and maybe extra objectives like minimizing risk or maximizing feature value variety. These objectives frequently clash, rendering it unattainable to optimize them concurrently to their maximum capacity. The proposed approach involves identifying a collection of Pareto-optimal solutions, whereby no solution within the collection is superior to another in all goals, ensuring non-dominance.In a multi-objective optimization problem, the goal is to optimize k objective functions simultaneously. Formally, it can be defined as:

$$\square\square\square\square(\square) = (\square_1(\square), \square_2(\square), \ldots, \square_{\square}(\square))$$

subject to$\square \in \square$, where Xis the feasible solution space, the objective is to find a set of non-dominated solutions that form the Pareto front. A solution is non-dominated if no other solution is better in all objectives.

i. Initialization phase:

The ACO algorithm starts with the initialization phase. During this stage, the graph's pheromone levels $\square_{ij}$ on all edges ($\square$,) are set to a modest positive constant $\square_0$. The initial pheromone value signifies the fundamental attractiveness of choosing a specific feature. During this step, many crucial parameters are established: the number of ants $\square$, the significance of the pheromone trail $\square$, the significance of heuristic information $\square$, the pace at which the pheromone evaporates $\square$ and a factor that amplifies the pheromone $\square$. These settings determine the behavior and effectiveness of the algorithm.

ii. Solution Construction Phase:

During the solution creation phase, each of the □ ants constructs a possible solution by probabilistically choosing features to include in the upcoming release. Every ant begins with an initial set of features □, which is empty, and gradually includes features using a decision algorithm that relies on probabilities. The likelihood The function $\square_{\square\square}(\square)$ represents the probability of an ant transitioning from a feature $\square_{\square}$ to feature $\square_{\square}$ at time □.

$$\square_{\square\square}(\square) = \frac{[\square_{ij}(\square)]^{\square}[\square_{ij}]^{\square}}{\sum_{\square\square\square\backslash\square}[\square_{ij}(\square)]^{\square}[\square_{ij}]^{\square}}$$

This formula $\square_{ij}(\square)$ represents the pheromone level on the edge connecting features $\square_{\square}$ and $\square_{\square}$, while $\square_{ij}$ represents the heuristic information associated with moving to feature $\square_{\square}$. A common heuristic is the benefit-to-cost ratio $\frac{\sum_{\square\square\square}\square_{\square\square}}{\square_{\square}}$, where $\square_{\square\square}$ is the benefit of the feature $\square_{\square}$ to customer $\square_{\square}$ and $\square_{\square}$ is the cost of implementing a feature $\square_{\square}$. The parameters □and □ control the relative importance of pheromone versus heuristic information.

iii. Multi-Objective Heuristic Information:

Define heuristic information $\square_{ij}$ considering multiple objectives. For example, if minimizing both cost and time, the heuristic could be:

$$\square_{ij} = \frac{1}{\square_1\square_{\square\square} + \square_2\square_{\square\square}}$$

where $\square_1$ and $\square_2$ are weights, and $\square_{\square\square}$ and $\square_{\square\square}$ are cost and time, respectively.

Each ant individually chooses features while ensuring that the combined cost of the selected features remains within the predetermined budget □. This feasibility assessment is essential to verify the validity of the implemented solution. The ant persists in incorporating more features until it is no longer possible to add more features without breaching the budget restriction.

iv. Pheromone Update Phase:

After all ants have completed their solutions, the pheromone levels on the edges are adjusted. This phase has two primary stages: pheromone dissipation and pheromone application.Pheromone evaporation is the gradual reduction of pheromone levels on all edges, imitating the natural dissipation over time. Mathematically, this may be expressed as:

$$\square_{ij}(\square + 1) = (1 - \square)\square_{ij}(\square)$$

where □ is the evaporation rate ($0<\square < 1$). This step prevents the algorithm from converging too quickly to a suboptimal solution by reducing the influence of previously laid pheromones. Next, each ant deposits pheromone on the edges it traversed, with the amount inversely proportional to the solution's total benefit:

$$\square_{ij}(\square + 1) = \square_{ij}(\square) + \sum_{\square=1}^{\square}\Delta\square_{ij}{}^{\square}$$

$$\Delta\square_{ij}{}^{\square} = \begin{cases} \frac{\square}{\square_{\square}} \ \square\square\ \square\square\square\ \square\ \square\square\square\square\ \square\square\square\square\ (\square,\square) \\ 0\ \square\square\square\square\square\square\square\square\square \end{cases}$$

where $\square_{\square}$ is the total benefit of the solution by ant□.

This iterative process of solution construction and pheromone updating continues for a predefined number of iterations or until convergence. Through these iterations, ants collectively explore the solution space, balancing exploiting known good solutions and exploring new possibilities, gradually leading to an optimal or near-optimal solution for the next release problem.

v. Iterative Optimization

The process of constructing the solution and updating the pheromone levels is continued for a predetermined number of iterations or until certain convergence conditions are satisfied. Throughout this iterative process, ants collaboratively investigate the solution space, maintaining a balance between strengthening the search around established effective solutions and diversifying to uncover new possible solutions. MO-ACO successfully navigates the complicated trade-offs in multi-objective optimization by using the interplay between exploration and exploitation.

vi. Storing and Returning Pareto-Optimal Solutions

During each cycle, the ants discover non-dominated solutions and store them in an external archive. The archive showcases the dynamic Pareto front, offering various solutions that effectively balance numerous objectives. Upon completion of the optimization process, the ultimate collection of Pareto-optimal solutions is provided, presenting decision-makers with various trade-off possibilities for the upcoming software release.

MO-ACO is a modified version of the traditional ACO framework that effectively addresses the multi-objective aspect of the Next Release Problem. MO-ACO systematically explores the solution space by repeatedly constructing, evaluating, and modifying pheromones. This approach allows it to uncover many Pareto-optimal solutions that address several objectives. This systematic methodology allows software planners to make well-informed decisions, balancing several factors to deliver the most ideal software releases.

Differential Evolution (DE)

Differential Evolution (DE) is an effective evolutionary technique for global optimization in continuous areas. It resolves optimization issues with non-linear, non-convex, or noisy objective functions. DE can be successfully incorporated with MO-ACO in the Next-Release Problem (NRP) to improve the software feature selection process for next-release planning. Below is a detailed explanation of Differential Evolution (DE) and its integration with Multi-Objective Ant Colony Optimization (MO-ACO):

i. Population Initialization:

Differential Evolution (DE) starts by initializing a population of potential solutions. Each answer □□ is denoted as a vector within the search space:

$$□_□ = (□_{□1}, □_{□2}, \ldots □_{□□})$$

where d is the number of dimensions in the problem space; typically, these solutions are created randomly while adhering to the problem's requirements.

ii. Mutation

Mutation is an essential operator in DE that allows for exploring previously unexplored areas inside the search space. DE creates a mutant vector □□ for each potential solution □□ by perturbing the existing solutions in the population. The mutant vector □□ is precisely specified as:

$$□_□ = □_{□1} + □.(□_{□2} - □_{□3})$$

where □1,□2,r1, and□3 are distinct randomly selected indices from the population, and F is the mutation scaling factor (typically set between 0 and 2).

iii. Crossover:

Crossover merges data from the mutant vector $□_□$ and the target vector $□_□$ to generate a trial vector $□_□$. The trial vector $□_□$ is generated by a binomial crossover operation:

$$□_{□□} = \begin{cases} □_{□□} & □□\ □□□□□() \le □□ \text{ or } j = \text{rand}()\ (1, d) \\ □_{□□} & □□□□□□□□□ \end{cases}$$

where rand()generates a random number between 0 and 1, and CR is the crossover probability (typically set between 0 and 1).

iv. Selection

The trial vector $□_□$ competes with the target vector $□_□$ based on their fitness values. The selection process determines which vector survives to the next generation based on a comparison of their objective function values□(□_□)and □(□_□):

$$□_{□□}^{□+1} = \begin{cases} □_{□□}, & □□\ □(□_□) < □(□_□) \\ □_{□□}, & □□□□□□□□□ \end{cases}$$

The MO-ACO algorithm is employed as a first step to systematically investigate and create a wide range of solutions for the Next-Release Problem. This algorithm effectively manages the trade-off between several competing objectives, such as optimizing customer happiness while minimizing development expenses. DE enhances MO-ACO by enhancing the solutions created during its exploration phase.

Experimental Evaluation

Dataset Description

Greer and Ruhe introduced the initial dataset employed [25]. The dataset comprises five customers and 20 requirements. Each requirement incurs a cost ranging from 1 to 10. The needs of each client are assigned a numerical value ranging from 1 to 5. Table 1 displays the cost, quantities, and relation among the requirements. Table 2 displays the significance of the consumers.

	$\square_1$	$\square_2$	$\square_3$	$\square_4$	$\square_5$	$\square_6$	$\square_7$	$\square_8$	$\square_9$	$\square_{10}$	$\square_{11}$	$\square_{12}$	$\square_{13}$		$\square_{14}$	$\square_{15}$	$\square_{16}$	$\square_{17}$	$\square_{18}$	$\square_{19}$	$\square_{20}$
C1	4	2	1	2	5	5	2	4	4	4	2	3	4		2	4	4	4	1	3	2
C2	4	4	2	2	4	5	1	4	4	5	2	3	2		4	4	2	3	2	3	1
C3	5	3	3	3	4	5	2	4	4	4	2	4	1		5	4	1	2	3	3	2
C4	4	5	2	3	3	4	2	4	2	3	5	2	3		2	4	3	5	4	3	2
C5	5	4	2	4	5	4	2	4	5	2	4	5	3		4	4	1	1	2	4	1
Cost	1	4	2	3	4	7	10	2	1	3	2	5	8		2	1	4	10	4	8	4

Table 2: Motorola datasetcustomer weight

	C1	C2	C3	C4	C5
Greer & Ruhe	1	4	2	3	4
Motorola	1	1	1	1	-

The Motorola dataset [29] is the second dataset. The dataset has a total of four customers and 35 requirements. The customers refer to four mobile telephone carriers, each requiring a separate set of smartphone functionality. All clients have the same weight. Every demand is associated with a certain cost and income, measured as a score. Furthermore, we allocate each need to a client using a consistent distribution. Table 2 displays the significance of the consumers.Table 3 provides each demand's cost, income, and corresponding customer.

Table 3: Motorola datasetcustomer request for requirements, cost, and revenue.

Customers	Requirements	Cost	Revenue
C4	$\square_1$	100	3
C2	$\square_2$	50	3
C3	$\square_3$	300	3
C1	$\square_4$	80	3
C4	$\square_5$	70	3
C4	$\square_6$	100	3
C3	$\square_7$	1000	3
C1	$\square_8$	40	3
C3	$\square_9$	200	3
C4	$\square_{10}$	20	1
C3	$\square_{11}$	1100	3
C1	$\square_{12}$	10	3
C3	$\square_{13}$	500	3
C3	$\square_{14}$	10	1
C4	$\square_{15}$	10	3
C1	$\square_{16}$	10	2
C2	$\square_{17}$	20	1
C2	$\square_{18}$	200	1
C1	$\square_{19}$	1000	3
C4	$\square_{20}$	120	2
C2	$\square_{21}$	300	2
C3	$\square_{22}$	50	1
C2	$\square_{23}$	10	2
C2	$\square_{24}$	30	3
C2	$\square_{25}$	110	2
C1	$\square_{26}$	230	2
C1	$\square_{27}$	40	1
C3	$\square_{28}$	180	2
C4	$\square_{29}$	20	2
C4	$\square_{30}$	150	2
C2	$\square_{31}$	60	3

C3	□32	100	1
C1	□33	400	3
C2	□34	80	1
C1	□35	40	1

Evaluation Measures

The efficacy of the proposed method's solutions was by utilizing three indicators to analyze the quality of multi-objective optimization problem solutions. This enables the identification of the effectiveness of the outcomes obtained by the MO-ACO-DE algorithm. The quality of the proposed solution was validated using the following metrics. The hyper-volume (HV) metric was employed to quantify the convergence and variety of the Pareto front [31]. The spread metric was employed to assess the variety of the solutions in the Pareto front and the number of non-dominated solutions (NDS) identified by the algorithm.

i. Hypervolume (HV): is used to calculate the total volume required by the fuzzy members in the objective space for a next release requirement queue (Q). A hypercube □□□ is generated for every requirement queue □ ∈ □, and the union of all the hypercubes forms HV. The mathematical formula for HV is as follows

$$□□ = □(\bigcup_{□=1}^{|□|} □□_□)$$

ii. Spread (Δ): is used to measure the spread extent of a set, and it is computed as follows

$$Δ = \frac{□_□ + □_□ + \sum_{□=1}^{□-1}|□_□ - \overline{□}|}{□_□ + □_□ + (□ - 1)\overline{□}}$$

Euclidean distance between consecutive solution queues is represented by □□, and their mean is represented by $\overline{□}$. The extreme Euclidean distances of the optimal PFS are measured by □□ □□□ □□.

iii. Non-Dominated Solutions (NDS): "Non-dominated" refers to better solutions than all other options in every objective. The proposed method was used to calculate the number of NDS. It is recommended that Pareto fronts be used, which contain a higher number of NDS.

EXPERIMENTAL RESULTS

Tables 4 and 5 present the outcome of the proposed model, MO-ACO-DE, for the HV indicator on two datasets. The results are compared with those produced by existing techniques and reflect the mean and standard deviation of the HV matrix for both datasets across four budget limits. The HV indicator demonstrates that as the value increases, the output quality also improves. Thus, the MO-ACO-DE method consistently achieves better results for every NRP instance in all scenarios.

Table 4: The mean and standard deviation of the HV measure for the four occurrences of the Greer & Ruhe dataset are being requested.

Algorithm Cost boundary		GRASP	NSGA-II	ACO	MOABC	MO-ACO-DE
Mean ± Std. dev	30%	7.708%±0. 37	9.015%±1.12	10.28%±6.57e-2	41.88%±1.15e 5	47.35%±1.24e-3
	50%	19.11%±0.350	20.65%±1.60	23.91%±6.75e-2	54.72%±2.64	58.84%±1.62e-3
	70%	32.24%±0.496	32.16%±2.30	38.46%±7.08e-2	60.86% ± 9.49e-4	62.57 ± 1.83e-3
	100%	-	-	-	62.67% ± 0.22	68.42 ± 1.97e-3

Table 5: The mean and standard deviation of the HV measure for the four occurrences of the Motorola dataset are being requested.

Algorithm Cost boundary		GRASP	NSGA-II	ACO	MOABC	MO-ACO-DE
Mean ± Std. dev	30%	4.088%±8.55e-3	7.920%±2.49e-1	8.517%±6.21e-2	41.23% ± 1.14e-2	44.29%±0.026
	50%	15.454% ±6.88e-2	18.006%±5.20e-1	19.159% ± 9.94e-2	51.12%±1.17e-2	59.16%±0.014
	70%	27.943% ±7.50e-2	31.710%±8.92e-1	32.777% ±1.14e-1	58.25%±7.00e-3	63.53%±6.42e-3

	100%	-	-	-	61.70%±4.94e 3	65.13 ± 1.77e-4

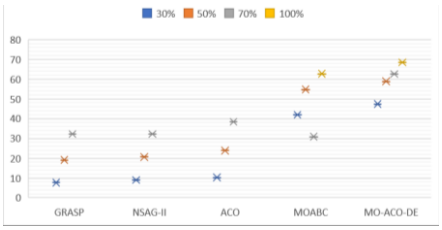

Figure 1:HV and iteration forGreer & Ruhe dataset

This indicates that the suggested method successfully balances exploring new possibilities, using known solutions, and converging toward optimal solutions within the search space. This is achieved by synergistically integrating the capabilities of the ACO and DE algorithms, facilitating the production of a more extensive array of non-dominated solutions (NDS) on the Pareto front, which are considered optimal for the NRP. Moreover, the outcomes from the suggested model exhibit minimal fluctuations across various budget constraints, suggesting that the overall enhancement attained by the model about this multi-objective measure is quite noteworthy. In contrast, the GRASP algorithm produces less accurate results due to its constrained and self-centered exploration. Unlike previous algorithms, this method does not consider a population, which restricts its ability to explore the search space efficiently. Figure 1 and Figure 2 depict the relationship between the average hyper-volume and the iteration number of the MO-ACO-DE technique on the Greer & Ruhe and Motorola datasets, respectively. The study is performed on both datasets using four alternative cost constraints (30%, 50%, 70%, and 100%).

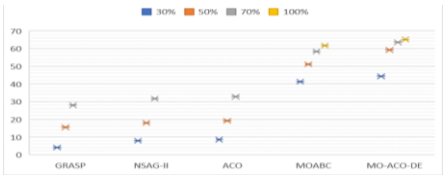

Figure 2:HV and iteration for Motorola dataset

Tables 7 and 8 display the mean and variability acquired using the D-Spread indicator. The findings of the MO-ACO-DE method are compared to those produced by existing algorithms using the two datasets. Smaller values of the Δ-spreadindicator correspond to superior outcomes. The Tables illustrate that the MO-ACO-DE algorithm yields the most optimal outcomes, demonstrating that this method is the preferred option for computing Pareto fronts with the most favorable solutions distribution throughout the dataset. The MO-ACO-DE algorithm explores all solutions through local and global searches and stores them in the archive. Consequently, all of the high-quality solutions generated by this procedure are retained. By transforming these optimum or non-dominated solutions into Pareto fronts, the algorithm becomes more diverse than the techniques that have been previously described. The numerical findings of the HV and D-Spread metrics indicate that our approach has superior outcomes. Figure 3 and Figure 4 illustrate the correlation between the Δ-spread and the iteration number of the MO-ACO-DE method on the Greer & Ruhe and Motorola datasets, respectively.

Table 5: The mean and standard deviation of the Δ-spread measure for the four occurrences of the Greer & Ruhe dataset.

Algorithm Cost boundary		GRASP	NSGA-II	ACO	MOABC	MO-ACO-DE
Mean ± Std. dev	30%	0.64±0.09	0.76±0.09	0.52±0.03	0.52±0.01	0.49 ± 0.002
	50%	0.73±0.07	0.79±0.07	0.52±0.01	0.48±0.01	0.43 ± 0.011
	70%	0.69±0.06	0.80±0.07	0.48±0.02	0.43±0.01	0.42 ± 0.01
	100%	-	-	-	0.39±0.05	0.36 ± 0.015

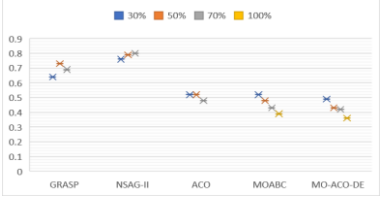

Figure 3: Δ-spread and iteration for Greer & Ruhe dataset

Table 6: The mean and standard deviation of the Δ-spread measure for the four occurrences of the Motorola dataset.

Algorithm Cost boundary		GRASP	NSGA-II	ACO	MOABC	MO-ACO-DE
Mean ± Std. dev	30%	0.60±0.04	0.80±0.07	0.68±0.06	0.45±0.02	0.38 ± 0.029
	50%	0.74±0.04	0.81±0.06	0.66±0.06	0.42±0.02	0.36 ± 0.011
	70%	0.70±0.03	0.77±0.05	0.61±0.06	0.38±0.02	0.34 ± 0.017
	100%	-	-	-	0.35±0.03	0.33 ± 0.013

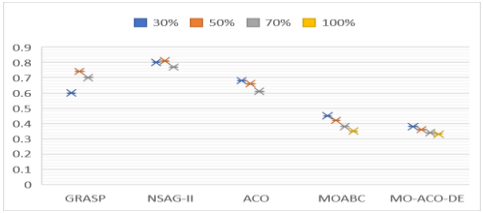

Figure 4: Δ-spread and iteration for Motorola dataset

The main aim of multi-objective optimization is to determine the Pareto front that contains the maximum number of solutions. As the number of options rises, the decision-maker's work of choosing the best one becomes easier. The optimum solutions for the NRP are non-dominated solutions rather than perfect solutions. Tables 7 and 8 present the mean and variability of non-dominated solutions produced by the MO-ACO-DE algorithm and other approaches discussed in prior studies for the two datasets. The results suggest that the MO-ACO-DE algorithm produces improved outcomes with more non-dominated solutions (NDS). The MO-ACO-DE technique allows for a thorough search space exploration, making it possible to identify all non-dominated solutions (NDS). Therefore, this approach can reveal more non-dominated solutions in the Pareto front. The Motorola dataset's NDS (Network Data Sets) have bigger dimensions due to their intricate nature and substantial magnitude. Hence, it is clear that the differences between MO-ACO-DE and the other published techniques are more noticeable for the two datasets. The Pareto front produced by the MO-ACO-DE algorithm for both datasets is illustrated in Figure 5 and Figure 6 correspondingly. Generally, this strategy is expected to be more efficient in exploring this issue's solution space than other strategies. Furthermore, an even distribution of non-dominated solutions (NDS) on the Pareto front, along with a higher quantity of NDS and superior hypervolume (HV) metrics, indicates that this approach surpasses others in performance.

Table 7: The mean and standard deviation of the NDS measure for the four occurrences of the Greer & Ruhe dataset.

Algorithm Cost boundary		GRASP	NSGA-II	ACO	MOABC	MO-ACO-DE
Mean ± Std. dev	30%	11.37±1.47	9.69±2.09	13.66±13.66	15.00±0.00	24.3±0.0130
	50%	17.65±2.22	11.30±1.82	17.75±0.61	23.66±0.48	31.4±0.3221
	70%	20.26±2.18	11.70±1.90	20.57±20.57	32.35±0.99	38.56±0.018
	100%	-	-	-	40.55±1.25	46.1±0.7504

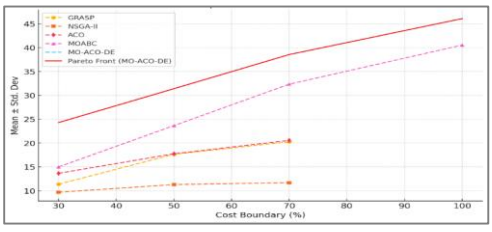

Figure 5: Pareto fronts generated on the Greer & Ruhe dataset

The Pareto front comparison for the Greer & Ruhe dataset in represented in Figure 5. The MO-ACO-DE algorithm consistently achieves higher values across all cost boundaries, highlighting its superior performance. The plot shows the performance of different algorithms under varying cost constraints, with MO-ACO-DE outperforming the others.

Table 8: The mean and standard deviation of the NDS measure for the four occurrences of the Motorola dataset.

Algorithm Cost boundary		GRASP	NSGA-II	ACO	MOABC	MO-ACO-DE
Mean ± Std. dev	30%	57.99±3.66	54.34±8.51	47.12±5.44	125.37±7.57	140.18±3.89
	50%	75.81±5.81	65.54±11.86	57.68±5.69	135.93±9.60	146.31±4.16
	70%	120.14±7.27	83.32±10.52	57.68±5.69	139.31±9.93	161.14±4.59
	100%	-	-	-	147.51±9.90	165.53±4.25

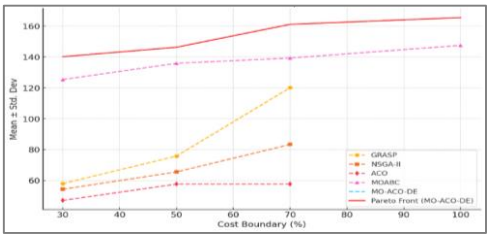

Figure 6: Pareto fronts generated on Motorola dataset

The MO-ACO-DE algorithm consistently obtains greater values across all cost limits, highlighting the Pareto front. Figure 6 illustrates the comparative performance of several algorithms under different cost limitations, with MO-ACO-DE demonstrating superior performance compared to the others.

CONCLUSION AND FUTURE WORK

This work introduces a novel method, Multi-Objective Ant Colony Optimisation with Differential Evolution (MO-ACO-DE), applied to two datasets. This technique was employed to address the constrained multi-objective version of the NRP. Integrating differential evolution (DE) algorithm operators into the MO-ACO algorithm boosts the performance of the MO-ACO-DE process, leading to enhanced solution accuracy and faster convergence time. Furthermore, the technique surpasses current multi-objective evolutionary algorithms regarding efficiency and effectiveness. The suggested method performs better than previously examined algorithms, including GRASP, NSGA-II, AOC, and MOABC. This advantage is evidenced by its greater convergence, variety, spread, and quantity of non-dominated solutions on two datasets about the problem. The findings demonstrate that the suggested algorithm generates a superior set of non-dominated solutions (NDS) located in the Pareto front compared to current approaches. These solutions display a narrower distribution and a greater hyper-volume. The MO-ACO-DE approach efficiently produces sets of requirements or Pareto fronts. These information sets assist software developers in making decisions and determining the optimal balance between requirements for the upcoming product release.

Future work will focus on extending the MO-ACO-DE approach to other complex multi-objective optimization problems beyond the NRP. Additionally, integrating other evolutionary algorithms with MO-ACO-DE will be explored to further enhance solution quality and convergence speed. The method's scalability will be tested on larger datasets and more diverse problem domains. Finally, adaptive mechanisms to dynamically adjust parameters within the algorithm will be developed to improve its robustness and applicability in real-world scenarios, aiding decision-making in software engineering and other fields.

REFERENCES

[1] G. Brau, J. Hugues, N. Navet, Towards the systematic analysis of non-functional properties in Model-Based Engineering for real-time embedded systems, Sci. Comput. Program. 156 (2018) 1–20.

[2] J. Jia, X. Yang, R. Zhang, X.i. Liu, Understanding software developers' cognition in agile requirements engineering, Sci. Comput. Program. 178 (2019) 1–19.

[3] J. Del Sagrado, I. M. Del Aguila, F. J. Orellana, and S. Tunez, Requirements selection: Knowledge based optimization techniques for solving the next release problem, In: 6th Workshop on Knowledge Engineering and Software Engineering (KESE 2010), Karlsruhe, Germany, 2010, pp. 40– 51.

[4] Jalali S M J, Ahmadian S, Khosravi A, Mirjalili S, Mahmoudi M R, Nahavandi S (2020) Neuroevolution-based autonomous robot navigation: a comparative study. CognSyst Res

[5] Maleki M, Wraith D, Mahmoudi MR, Contreras-Reyes JE (2020) Asymmetric heavy-tailed vector auto-regressive processes with application to financial data. J Stat Comput Simul 90(2):324–340.

[6] Heydari MH, Atangana A, Avazzadeh Z, Mahmoudi MR (2020) An operational matrix method for nonlinear variable-order time fractional reaction–diffusion equation involving Mittag-Leffler kernel. The Europ Phys J Plus 135(2):1–19.

[7] Soltani AR, Nematollahi AR, Mahmoudi MR (2019) On the asymptotic distribution of the periodograms for the discrete time harmonizable simple processes. Stat Infer Stoch Process 22(2): 307–322.

[8] Heydari MH, Avazzadeh Z, Mahmoudi MR (2019) Chebyshev cardinal wavelets for nonlinear stochastic differential equations driven with variable-order fractional Brownian motion. Chaos, Solitons & Fractals 124:105–124. https://doi.org/10.1016/j.chaos. 2019.04.040.
[9] Maleki M, Contreras-Reyes JE, Mahmoudi MR (2019) Robust mixture modeling based on two-piece scale mixtures of normal family. Axioms 8(2):38 . Zarei AR, Shabani A, Mahmoudi MR (2019) Comparison of the climate indices based on the relationship between yield loss of rainfed winter wheat and changes of climate indices using GEE model. Sci Total Environ 661:711–722.
[10] Bagherinia A, Minaei-Bidgoli B, Hossinzadeh M, Parvin H (2019) Elite fuzzy clustering ensemble based on clustering diversity and quality measures. Appl Intell 49(5):1724–1747.
[11] Mojarad M, Nejatian S, Parvin H, Mohammadpoor M (2019) A fuzzy clustering ensemble based on cluster clustering and iterative fusion of base clusters. Appl Intell 49(7):2567–2581.
[12] Parvin H, Nejatian S, Mohamadpour M (2018) Explicit memory based ABC with a clustering strategy for updating and retrieval of memory in dynamic environments. Appl Intell 48(11):4317–4337.
[13] Niu H, Khozouie N, Parvin H, Alinejad-Rokny H, Beheshti A, Mahmoudi MR (1891) An Ensemble of Locally Reliable Cluster Solutions. Appl Sci 10(5):2020.
[14] Pan JJ, Mahmoudi MR, Baleanu D, Maleki M (2019) On comparing and classifying several independent linear and non-linear regression models with symmetric errors. Symmetry 11(6):820.
[15] Mahmoudi MR, Mahmoudi M, Pak A (2019) On comparing, classifying and clustering several dependent regression models. J Stat Comput Simul 89(12):2280–2292.
[16] Mahmoudi MR, Heydari MH, Roohi R (2019) A new method to compare the spectral densities of two independent periodically correlated time series. Math Comput Simul 160:103–110.
[17] Mahmoudi MR, Heydari MH, Avazzadeh Z (2019) Testing the difference between spectral densities of two independent periodically correlated (cyclostationary) time series models. Commun Stat Theory Methods 48(9):2320–2328.
[18] Durillo J J, Zhang Y, Alba E, Nebro A J (2009) A study of the multi-objective next release problem. In search based software engineering, 2009 1st international symposium on (pp. 49-58). IEEE
[19] Sajjad U, Hanif M Q (2010) Issues and Challenges of Requirement Elicitation in Large Web Projects (Dissertation). Retrieved from http://urn.kb.se/resolve?urn=urn:nbn:se:bth-3216.
[20] Del Sagrado J, Del Aguila IM, Orellana FJ (2015) Multi-objective ant colony optimization for requirements selection. EmpirSoftw Eng 20(3):577–610.
[21] Zhang Y, Harman M (2010) Search based optimization of requirements interaction management. In 2nd international symposium on search based software engineering (pp. 47-56). IEEE.
[22] Almeida J C, Pereira F D C, Reis M V, Piva B (2018) The next release problem: complexity, exact algorithms and computations. In international symposium on combinatorial optimization (pp. 26- 38). Springer, Cham.
[23] Coello CAC, Lamont GB, Van Veldhuizen DA (2007) Evolutionary algorithms for solving multi-objective problems (Vol. 5). Springer, New York.
[24] Taradeh M, Mafarja M, Heidari AA, Faris H, Aljarah I, Mirjalili S, Fujita H (2019) An evolutionary gravitational search-based feature selection. Inf Sci 497:219–239.
[25] Faris H, Al-Zoubi AM, Heidari AA, Aljarah I, Mafarja M, Hassonah MA, Fujita H (2019) An intelligent system for spam detection and identification of the most relevant features based on evolutionary random weight networks. Inf Fusion 48:67–83.
[26] Gharehchopogh FS, Gholizadeh H (2019) A comprehensive survey: whale optimization algorithm and its applications. Swarm EvolComput 48:1–24.
[27] A.J. Bagnall, V.J. Rayward-Smith, I.M. Whittley, The next release problem, Inf. Softw. Technol. 43 (14) (2001) 883–890.
[28] N. Veerapen, G. Ochoa, M. Harman, E.K. Burke, An integer linear programming approach to the single and bi-objective next release problem, Inf. Softw. Technol. 65 (2015) 1–13.
[29] Y. Zhang, M. Harman, S.A. Mansouri, The multi-objective next release problem, in: Proceedings of the 9th annual conference on Genetic and evolutionary computation, London, England, United Kingdom, 2007, pp. 1129–1136.
[30] M. Ghasemi, K. Bagherifard, H. Parvin, S. Nejatian, K.-H. Pho, Multi-objective whale optimization algorithm and multi-objective grey wolf optimizer for solving next release problem with developing fairness and uncertainty quality indicators, Appl. Intell. 51 (8) (2021) 5358–5387.
[31] M.H. Marghny, H.M. El-Hawary, W.H. Dukhan, An effective method of systems requirement optimization based on genetic algorithms, Inform. Sci. Lett. 6 (1) (2017) 15–28.
[32] D. Greer, G. Ruhe, Software release planning: an evolutionary and iterative approach, Inf. Softw. Technol. 46 (4) (2004) 243– 253.
[33] A. Finkelstein, M. Harman, S.A. Mansouri, J. Ren, Y. Zhang, A search based approach to fairness analysis in requirement assignments to aid negotiation, mediation and decision making, Requirements Eng. 14 (4) (2009) 231–245.
[34] J.M. Chaves-Gonza´lez, M.A. Pe´rez-Toledano, A. Navasa, Teaching learning based optimization with Pareto tournament for the multiobjective software requirements selection, Eng. Appl. Artif. Intell. 43 (2015) 89–101.

[35] J.M. Chaves-Gonza´lez, M.A. Pe´rez-Toledano, A. Navasa, Software requirement optimization using a multiobjective swarm intelligence evolutionary algorithm, Knowl.-Based Syst. 83 (2015) 105–115.
[36] J.M. Chaves-Gonza´lez, M.A. Pe´rez-Toledano, Differential evolution with Pareto tournament for the multi-objective next release problem, Appl. Math. Comput. 252 (2015) 1–13.
[37] Aydemir F B, Dalpiaz F, Brinkkemper S, Giorgini P, Mylopoulos J (2018) The next release problem revisited: a new avenue for goal models. In 2018 IEEE 26th international requirements engineering conference (RE) (pp. 5-16). IEEE.
[38] Hamdy A, Mohamed AA (2019) Greedy binary particle swarm optimization for multi-objective constrained next release problem. Int J Mach Learn Comput 9(5):561–568.
[39] Mirjalili S, Saremi S, Mirjalili SM, Coelho LDS (2016) Multiobjective grey wolf optimizer: a novel algorithm for multicriterion optimization. Expert Syst Appl 47:106–119.
[40] R. Massobrio, S. Nesmachnow, F. Palomo-Lozano, B. Dorronsoro, Virtual Savant as a generic learning approach applied to the basic independent Next Release Problem, Appl. Soft Comput. 108 (2021) 107374.
[41] L. Li, M. Harman, E. Letier and Y. Zhang, Robust next release problem: handling uncertainty during optimization, In Proceedings of the 2014 Annual Conference on Genetic and Evolutionary Computation (2014, July), (pp. 1247–1254), ACM.
[42] A.A. Araujo, M. Paixao, I. Yeltsin, A. Dantas and J. ´ Souza, An architecture based on interactive optimization and machine learning applied to the next release problem, Automated Software Engineering 24(3) (2017), 623–671.
[43] J.-W. Ho, C.-C. Fang, and Y.-S. Huang, "The determination of optimal software release times at different confidence levels with consideration of learning effects," Softw. Test., Verification Rel., vol. 18, no. 4, pp. 221–249, Dec. 2008.
[44] K.-C. Chiu, J.-W. Ho, and Y.-S. Huang, "Bayesian updating of optimal release time for software systems," Softw. Qual. J., vol. 17, no. 1, pp. 99–120, Mar. 2009.
[45] V. Nagaraju and L. Fiondella, "Online optimal release time for nonhomogeneous Poisson process software reliability growth model," in Proc. Annu. Rel. Maintainability Symp. (RAMS), Jan. 2020, pp. 1–6.
[46] V. Kumar, V. B. Singh, A. Dhamija, and S. Srivastav, "Cost-reliability optimal release time of software with patching considered," Int. J. Rel., Qual. Saf. Eng., vol. 25, no. 4, Aug. 2018, Art. no. 1850018.
[47] J. Yang, J. Chen, and X. Wang, "EM algorithm for estimating reliability of multi-release open source software based on general masked data," IEEE Access, vol. 9, pp. 18890–18903, 2021.
[48] P. Prashant, A. Tickoo, S. Sharma, and J. Jamil, "Optimization of cost to calculate the release time in software reliability using Python," in Proc. 9th Int. Conf. Cloud Comput., Data Sci. Eng. (Confluence), Jan. 2019, pp. 471–474.
[49] Harman M, Krinke J, Medina-Bulo I, Palomo-Lozano F, Ren J, Yoo S (2014) Exact scalable sensitivity analysis for the next release problem. ACM Trans Softw Eng Methodol (TOSEM) 23(2):19–31. https://doi.org/10.1145/2537853
[50] Domínguez-Ríos MÁ, Chicano F, Alba E, del Águila I, del Sagrado J (2019) Efficient anytime algorithms to solve the bi-objective next release problem. J SystSoftw 156:217–231.
[51] P. Pirozmand, A. Ebrahimnejad, H. Alrezaamiri, H. Motameni, A novel approach for the next software release using a binary artificial algae algorithm, J. Intell. Fuzzy Syst. 40 (3) (2021) 5027–5041.
[52] T. Ferreira, S. Vergilio, M. Kessentini, 'Implementing SearchBased Software Engineering Approaches with Nautilus', Brazilian Sympos. Softw. Eng. (2021) 303–308.
[53] Song, B., & Wang, Z. (2016). A new genetic algorithm approach to smooth path planning for mobile robots. Assembly Automation, 36, 138–145.
[54] Song, B., Wang, Z., & Zou, L. (2017). On global smooth path planning for mobile robots using a novel multimodal delayed PSO algorithm. Cognitive Computation, 9, 5–17.
[55] Zuo, L., Guo, Q., Xu, X., & Fu, H. (2015). A hierarchical path planning approach based on A∗ and least-squares policy iteration for mobile robots. Neurocomputing, 170, 257–266.
[56] Y. Jin, T. Okabe, S. Sendhof, Adapting weighted aggregation for multiobjective evolution strategies, in: International Conference on Evolutionary Multi-Criterion Optimization. Springer, Berlin, Heidelberg, 2001, pp. 96–110.
[57] D. Kalyanmoy and H. Jain, "An evolutionary many-objective optimization algorithm using reference-point-based nondominated sorting approach, part I: solving problems with box constraints," IEEE Transactions on evolutionary computation, vol. 18, no. 4, pp. 577-601, 2013.