EUROPEAN JOURNAL OF PURE AND APPLIED MATHEMATICS
2025, Vol. 18, Issue 4, Article Number 7170
ISSN 1307-5543 – ejpam.com
Published by New York Business Global

Solving Multi-objective Resource Allocation Problem Using a Novel Optimization Approach: Genetic Algorithm with Hybrid Mutation

M.A. El-Shorbagy[1,*], M.A. Elsisy[2]

[1] *Department of Mathematics, College of Science and Humanities in Al-Kharj, Prince Sattam bin Abdulaziz University, Al-Kharj 11942, Saudi Arabia*
[2] *Department of Basic Engineering Science, Faculty of Engineering at Benha, Benha University, Benha, Egypt*

Abstract. The multi-objective resource allocation problem (MORAP) refers to the challenge of distributing limited resources across multiple projects or business divisions while simultaneously satisfying several, often conflicting, objectives. Such problems are common in engineering, management, and operations research, where decision-makers must balance cost, efficiency, and performance. To address this challenge, this paper introduces a novel Genetic Algorithm with Hybrid Mutation (GA-HM) specifically designed for MORAP. The proposed approach integrates two complementary mutation operators—displacement mutation and inversion mutation—applied in a randomized manner. This hybridization increases the exploration capability of the algorithm, maintains population diversity, and reduces the risk of premature convergence to local optima. To evaluate the effectiveness of GA-HM, two benchmark test problems from the literature are considered, both involving multi-objective workforce-task allocation. Experimental results clearly demonstrate that GA-HM consistently produces superior solutions compared to existing approaches such as fuzzy dynamic programming, fuzzy dynamic optimization, effective GA methods, k-means-based GA, and multi-objective hybrid GAs. Importantly, GA-HM not only identifies optimal trade-offs between cost and efficiency but also provides a well-distributed set of non-dominated (ND) solutions, offering decision-makers a broader range of alternatives for resource planning. Overall, the findings confirm that GA-HM is a robust and efficient method for solving MORAPs. By generating diverse Pareto-optimal solutions, the algorithm equips practitioners with practical decision support tools for complex multi-objective optimization tasks. Moreover, the proposed methodology demonstrates strong potential for extension to other real-world engineering and management applications that require effective multi-objective resource allocation strategies.

2020 Mathematics Subject Classifications: 90C29, 68W50, 90B30

Key Words and Phrases: Multi-objective optimization, resource allocation problem, genetic algorithm, optimization

---

*Corresponding author.
DOI: https://doi.org/10.29020/nybg.ejpam.v18i4.7170

Email addresses: mohammed_shorbagy@yahoo.com (M.A. El-Shorbagy)

1. Introduction

The purpose of resource allocation is to distribute available resources in a cost-effective and efficient manner in order to achieve organizational goals. It forms an essential component of resource management, as it ensures that tasks and projects are completed within defined constraints of time, manpower, and budget. Effective allocation requires careful planning of not only how much time is needed for each activity but also the types and quantities of resources required. This systematic distribution of resources is commonly referred to as resource allocation [1].

When extended to multi-objective settings, resource allocation becomes significantly more complex and is referred to as the Multi-Objective Resource Allocation Problem (MORAP). In this context, resources are distributed across multiple projects, departments, or tasks in order to achieve competing objectives. A resource, in general, may refer to manpower, raw materials, capital, equipment, or any limited asset that can be utilized to accomplish a goal. The objectives themselves are often conflicting—for instance, organizations may simultaneously seek to maximize profits, reduce costs, improve efficiency, or enhance overall quality [2].

MORAP is highly relevant in a wide range of practical applications. It plays a crucial role in the distribution of marketing resources [3], where marketing managers must allocate limited resources—such as advertising budgets, sales teams, store space, and product inventory—across different markets or products. In network optimization [4], MORAP contributes to improving the performance of stochastic network systems by balancing activity duration and resource utilization. It is also applied in workforce scheduling within manufacturing [5], where efficient allocation of workers helps control production workloads, reduce labor costs, prevent overwork, and shorten production cycles. In the field of financial program optimization [6], MORAP supports the development of investment portfolios by allocating capital among stocks, bonds, or other assets in a way that maximizes returns while minimizing risk. Furthermore, it has significant applications in healthcare resource allocation [7], ensuring cost-effectiveness and maximizing benefits through the effective distribution of staff, equipment, and funding in both clinical and public health settings. These examples highlight that MORAP is not limited to a single domain but can be applied to an endless variety of problems across industries and disciplines [8].

Traditionally, exact optimization methods such as Integer Programming (IP), Dynamic Programming, and Branch-and-Bound have been applied to solve MORAP [9]. While these techniques can yield precise solutions, they are computationally expensive and become impractical for large-scale, real-world problems due to the exponential growth of the solution space [10].

To overcome these challenges, researchers have increasingly adopted evolutionary algorithms (EAs) and other metaheuristic techniques that can efficiently explore complex search spaces and provide near-optimal solutions. Popular approaches include Genetic Algorithms (GAs) [11], fuzzy multi-objective GAs [12], Monte Carlo Tree Search (MCTS) [13], Ant Colony Optimization (ACO) [14], Variable Neighborhood Search [15], Memetic Algorithms based on node-weighted graphs [16], and Particle Swarm Optimization (PSO)

[17].

Among these approaches, Genetic Algorithms (GAs) stand out due to their robustness, adaptability, and global search capabilities. Originating from principles of natural selection, genetics, and evolution, the GA was first introduced in 1975 [18] and later formalized in 1989 [19] as a powerful method for addressing large-scale optimization problems. Since then, GAs have gained significant attention in the academic community and have been widely applied to multi-objective optimization. Research consistently shows that GAs are particularly effective in solving MORAP, as they are capable of producing diverse sets of Pareto-optimal solutions, thereby offering decision-makers flexibility in balancing multiple conflicting objectives.

This paper introduces a Genetic Algorithm with Hybrid Mutation (GA-HM) as a novel approach for addressing the Multi-Objective Resource Allocation Problems (MORAPs). The distinguishing feature of GA-HM lies in its hybrid mutation stage, which integrates multiple mutation strategies to preserve and introduce diversity into the evolving population. By maintaining genetic variation, GA-HM prevents the population from becoming too homogeneous, thereby reducing the likelihood of premature convergence and enabling the algorithm to continue exploring new areas of the solution space. This mechanism is particularly important for avoiding entrapment in local minima, which can otherwise halt or significantly slow the evolutionary process.

The performance of GA-HM was validated through simulation experiments on multiple benchmark test problems drawn from the literature. The results consistently demonstrate that GA-HM achieves superior outcomes compared to existing methods, highlighting its effectiveness and robustness in solving MORAP. The hybrid mutation mechanism not only enhances the algorithm's search capability but also ensures a more balanced exploration–exploitation trade-off, which is critical in multi-objective optimization.

A key challenge in MORAP, as with many real-world optimization tasks, is that objectives are frequently conflicting. For instance, minimizing cost often comes at the expense of efficiency or quality, and maximizing one performance metric may require compromising another. As a result, such problems rarely yield a single globally optimal solution. Instead, the outcome is typically a set of trade-off solutions that represent different balances among objectives. These solutions are referred to as non-dominated solutions or Pareto-optimal solutions [20].

A solution is considered Pareto-optimal if no other solution exists that can improve one objective without worsening at least one of the others. Collectively, these solutions form the Pareto front, which provides decision-makers with a diverse range of alternatives to choose from, depending on their priorities and constraints. The strength of GA-HM lies in its ability to approximate this Pareto front effectively, generating a well-distributed set of high-quality solutions. This ensures that stakeholders are equipped with a flexible decision-support tool that allows for informed trade-offs across multiple objectives.

The remainder of this article is structured as follows. Section 2 provides the theoretical foundation of the Multi-Objective Resource Allocation Problem (MORAP), including its formal definition, applications, and the challenges associated with solving it. Section 3 presents an overview of the Genetic Algorithm (GA), outlining its fundamental components such as selection, crossover, and mutation, as well as its suitability for multi-objective optimization. Section 4 introduces the proposed Genetic Algorithm with Hybrid Mutation (GA-HM) in detail, describing the hybrid mutation strategy, its role in preserving diversity. Section 5 reports and analyzes the experimental findings obtained from benchmark test problems, comparing GA-HM with other state-of-the-art methods and discussing its effectiveness in solving MORAP. Section 6 explains GA-HM's computational complexity. Finally, Section 7 concludes the paper by summarizing key insights, highlighting contributions, and suggesting possible directions for future research.

2. Multi-Objective Resource Allocation Problem

The following is a mathematical expression for a general minimization problem of Q objectives:

Given

$$x = [x_1, x_2, \dots, x_n],$$

where n is the dimension of the decision variable space,

$$\min / \max \; F(x) = [f_q(x)], \quad q = 1, \dots, Q,$$

subject to

$$g_j(x) \le 0, \quad j = 1, \dots, J,$$

where $f_q(x)$ is the q-th objective function and $g_j(x)$ is the j-th inequality constraint. The multi-objective optimization (MOO) problem then reduces to finding x such that

$$f_q(x), \quad \forall q = 1, \dots, Q$$

is optimized.

Pareto dominance is a notion used in MOO to evaluate the solutions. The definition of this Vilfredo Pareto–formulated idea is [3].

Definition 1 (Pareto dominance). A vector

$$u = (u_1, u_2, \dots, u_Q)$$

is said to dominate a vector

$$v = (v_1, v_2, \dots, v_Q)$$

(denoted by $u \succ v$), for a minimization problem, if and only if

$$\forall q \in \{1, \dots, Q\} : \; u_q \le v_q \quad \wedge \quad \exists q \in \{1, \dots, Q\} : \; u_q < v_q.$$

2.1. Formulation of Multi-Objective Resource Allocation Problem

The general formulation of the multi-objective resource allocation problem (MORAP) is as follows [11]:

$$
\begin{aligned}
&\max\ Z_1(x_1, x_2, \ldots, x_n) = \sum_{k=1}^{n} z_k^1(x_k), \\
&\max\ Z_2(x_1, x_2, \ldots, x_n) = \sum_{k=1}^{n} z_k^2(x_k), \\
&\quad \vdots \\
&\max\ Z_q(x_1, x_2, \ldots, x_n) = \sum_{k=1}^{n} z_k^q(x_k), \\
&\text{subject to: } \sum_{k=1}^{n} g_k(x_k) \le s, \quad g_k(x_k) \ge 0, \quad x_k \ge 0.
\end{aligned}
\tag{1}
$$

where $x_k = [x_1, x_2, \ldots, x_n]$ are the decision variables and the available resources are denoted by s. $g_k(x_k)$ stands for the stages of the activity, while the objective function is $Z_q(x_1, x_2, \ldots, x_n)$. With a finite quantity of resources available, the MORAP process attempts to maximize the achievement of each objective by distributing those resources (decision variables) among several tasks (activity stages), each of which is represented by an activity stage [11].

MORAP appears in a wide range of practical domains where resources are limited and multiple criteria must be balanced, such as:

- Manufacturing and Production Planning,
- Project Management,
- Telecommunications and Networks,
- Healthcare Resource Allocation,
- Energy Systems,
- Finance and Investment.

MORAP has many challenges due to several reasons:

(i) **Multiple Conflicting Objectives:** Improving one objective often worsens another (e.g., increasing speed may increase cost). This requires finding a Pareto front rather than a single solution.

(ii) **Combinatorial Complexity:** Discrete resources or allocation units often lead to NP-hard problems, as the number of possible allocations grows exponentially with the number of resources and activities.

(iii) **Nonlinearity:** Objective functions may be nonlinear or nonconvex, making traditional optimization methods ineffective.

(iv) **Constraints:** Resource limits, operational restrictions, or policy requirements create complex feasible regions.

(v) **Dynamic and Uncertain Environments:** Resource availability or demand may change over time and require adaptive or stochastic approaches.

(vi) **Trade-off Analysis:** Decision-makers must select a final solution from the Pareto-optimal set, which can be subjective and context-dependent.

3. Genetic Algorithm Mechanism

Genetic algorithms (GAs) operate on a pool of potential solutions stored in a limited number of bits, or chromosomes. Each chromosome communicates with the others to find the best solution by utilizing operators taken from natural genetics [21, 22].

The following is a description of the stages involved in basic genetic operators:

Step 0: Create an initial population at random (the initialization stage).

Step 1: Determine the values of the Q objectives for each individual (evaluation of non-dominated solutions).

Step 2: In this step, utilize selection operators to choose which individuals will be used with the crossover operators.

Step 3: Use a crossover operator on each of the chosen pair of chromosomes by selection operator to produce a new offspring.

Step 4: Use the mutation operators to modify each offspring that is produced by the crossover process.

Step 5: Verify the stopping requirement. Go back to Step 1 if it is not fulfilled. Otherwise, proceed to step 6.

Step 6: The best solution (chromosome) is offered to the decision-maker (DM).

Figure 1 shows a flow diagram depicting the main steps of the GA process.

GA can be extended to address multiple conflicting objectives. In this context, solutions are assessed using Pareto dominance, resulting in a set of Pareto-optimal solutions rather than a single optimum. This approach offers several advantages for multi-objective optimization: it can handle complex, nonlinear, and discrete search spaces, maintain a diverse population to provide multiple trade-off solutions, and offers flexibility in incorporating constraints and hybrid strategies.

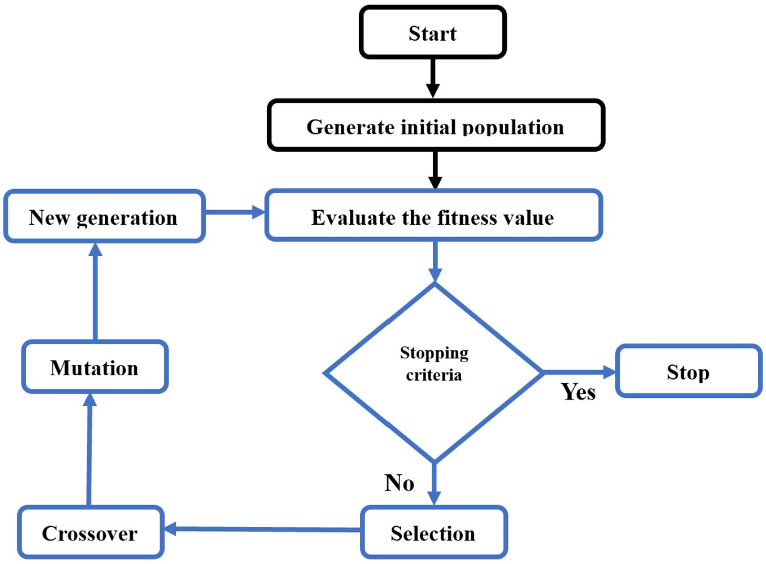

Figure 1: GA main flow diagram.

4. The Proposed Algorithm

We present the genetic algorithm with hybrid mutation (GA-HM) as a solution to MORAP in this study. During the hybrid mutation stage, diversity is maintained and introduced. This helps the algorithm avoid local minima when evolution is slowed or halted due to chromosomes population homogeneity. There are numerous stages to GA-HM. The following subsections elaborate on them.

Step 1: Initialization phase

The goal of MORAP is to discover the most efficient and least expensive route between two nodes, the source node (s) and the terminal node (t). The set of arcs

$$(s, x_{1m}), (x_{1m}, x_{2m}), \ldots, (x_{(N-1)m}, t),$$

constitutes routes from node s to node t. A path from node s to node t is a sequence of nodes

$$(s, x_{1m}), (x_{1m}, x_{2m}), \ldots, (x_{(N-1)m}, t).$$

The sequence of nodes can be thought of as an equivalent representation of a path, as seen in Figure 2, where N is the number of stages (districts) and $m = 1, 2, \ldots, m$ [23]. Consequently, the structure of each chromosome can be represented as a sequence of nodes (path).

Here, the algorithm creates a starting population P of $Npop$ chromosomes. According to Figure 3 elements are chosen at random from the available numbers in each district [24]. For instance, in MORAP assigning three workers to a specific set of four jobs, there are four stages. The goal of multi-objective resource allocation is to find the optimal four-stage allocation path while minimizing costs and maximizing profits. At each stage, we use a random selection element ranging from 0 to 3 to create an initial population [25]. Table 1 shows the four-stage allocation path.

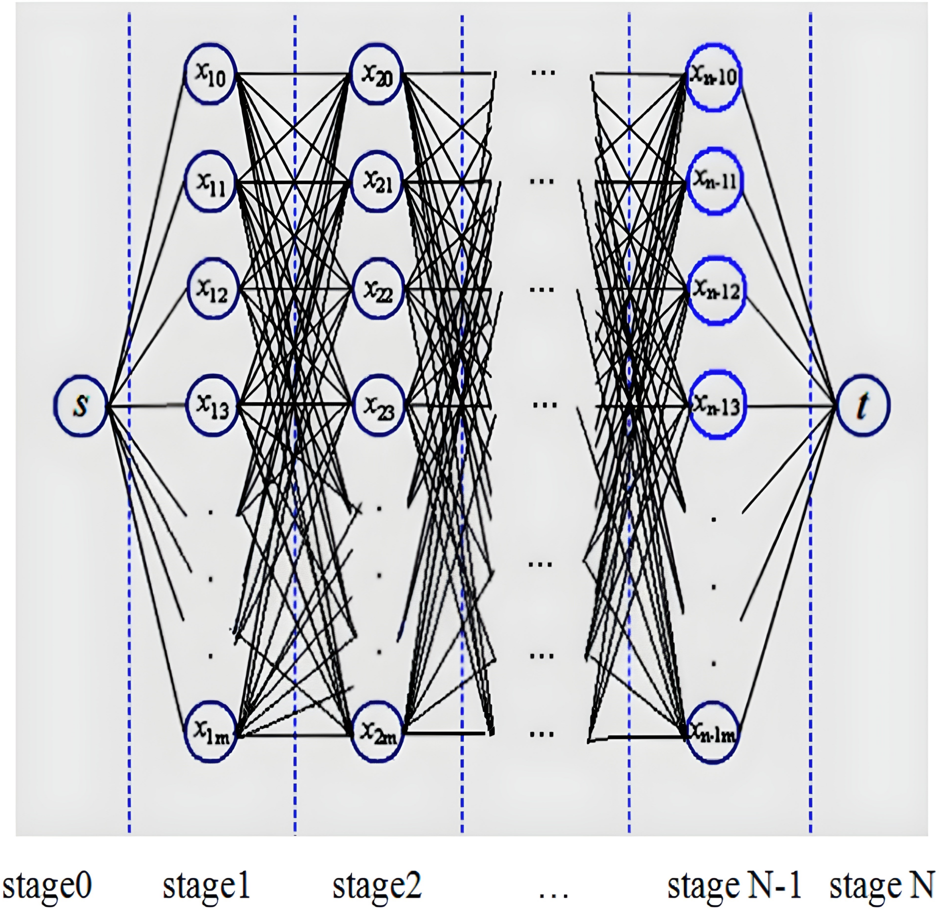

Figure 2: Analysis of MORAP as a network model.

Step 2: Assessment of Solutions

In this step, the goal function values for the produced strings are obtained. No optimal solution exists for a multi-objective optimization problem that aims to maximize all objectives at the same time. In such instances, there is a (potentially endless) number of solutions, and the goal functions are characterized as conflicting with one another. When a solution is non-dominated and Pareto optimum, we say that it exists [18]. Therefore, sorting a population by non-domination is essential for finding the Pareto optimal solutions.

Step 3: Categorizing the population based on their non-dominated solutions.

Table 1: The four-stage allocation path is based on chromosomal structure

Stage (Index)	1	2	3	4
Path Selection	0	1	2	3
The route for allocation	10	21	32	43

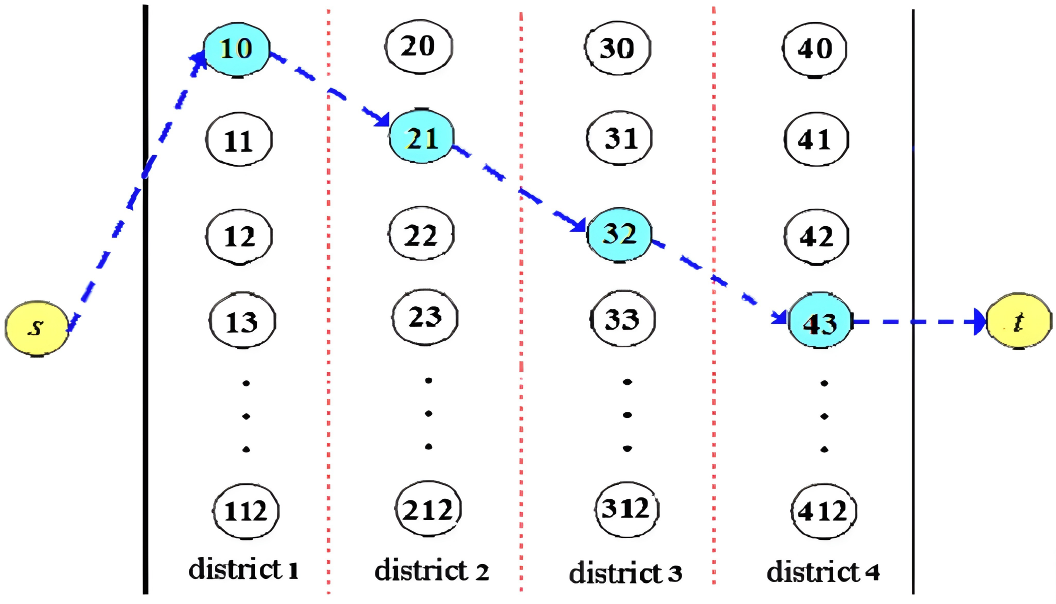

Figure 3: A diagram shows the best route for allocation.

Suppose there is a population whose members all have objective function values of q $(q > 1)$. To locate the set of solutions that are not dominated, follow this technique [26].

(i) Start with $R = 1$.

(ii) For all $P = 1, 2, \ldots, N_{\text{POP}}$ and $R \neq P$, the solution x_R is said to dominate the solution x_P if the two conditions are achieved:

- x_R is no worse than x_P in all objectives, or

$$f_q(x_R) \leq f_q(x_P), \quad \forall q = 1, \ldots, Q.$$

- The solution x_R is strictly better than x_P in at least one objective, or

$$f_q(x_R) < f_q(x_P), \quad \text{for at least one } q \in \{1, 2, \ldots, Q\}.$$

(iii) If for any P, x_P is dominated by x_R, mark x_P as “dominated”, and it is inefficient.

(iv) If all solutions (that is, when $R = N_{\text{POP}}$ is reached) in the set are considered, go to (v). Otherwise, increment R by one and go to (ii).

(v) The solutions which are not assigned “dominated” are assigned as non-dominated solutions.

The non-dominated front in the population of that generation is composed of all these non-dominated solutions. If the stopping criterion is reached, use the non-dominated solutions to proceed to DM; otherwise, continue to the following step.

Step 4: Stage of Selection

The primary goal of the selection phase, also known as the parent selection phase, is to identify the people with the greatest potential and pass their genes on to subsequent generations [24]. By increasing the likelihood that the best chromosomes will be passed down to future generations, the selection (reproduction) operator aims to raise the population's average quality [27]. The proposed algorithm uses a tournament selection method, in which N_s chromosomes, at random, are chosen from the population, and the chromosome with the highest fitness wins and is added to the population of the following generation.

Step 5: Operator of crossover

The purpose of chromosomal crossover is to transfer genetic information from one generation to the next [28]. After the selection process, $P_m \times N_s$ chromosomes are chosen for crossover. This study uses uniform crossover operator [29]. Here, a chromosome of uniform size is used to generate a random binary string. The next step is the exchange of relative genes inside this binary string between parents. The bits of the parent strings are swapped at the location in the random binary string that corresponds to 1. If this is not the case, then no bit exchange takes place [30]. Figure 4 illustrates how chromosomes from two parents can be crossed to create new chromosomes.

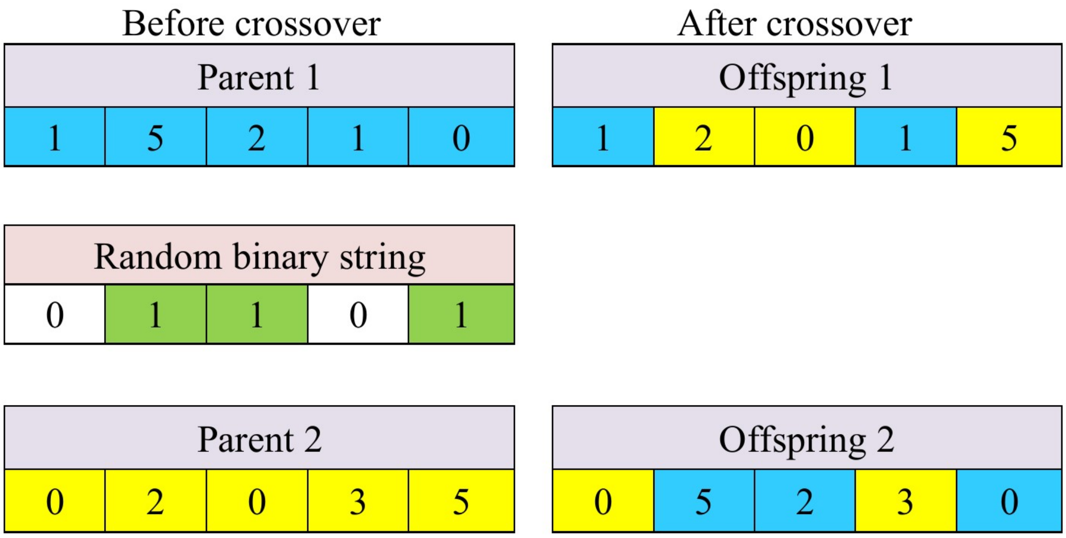

Figure 4: The process of crossover.

Step 6: The stage of hybrid mutation

Together, the displacement mutation operator and the inversion mutation operator [31] form the basis of our method. By providing variety while simultaneously retaining it, hybrid mutation helps the algorithm avoid local minima, which can halt or significantly impede development. So, after performing the crossover, only a proportion $N_m = [P_m \times N_c]$ of the resulting $N_c = [P_c \times N_s]$ offspring undergo mutation, where P_m is the mutation probability - fraction of offspring that will be mutated. The mutation operation is as follows:

(i) Set P_m and N_c, then compute $N_m = [P_m \times N_c]$.

(ii) Randomly select N_m offspring to mutate.

(iii) For each selected offspring, generate a random number $r \in [0, 1]$.

(iv) If $r > 0.5$ the displacement mutation is executed; otherwise, executed the inversion mutation.

- A substring of genes is chosen at random and placed in a random location in displacement mutation. Thus, it is possible to think of insertion as a particular instance of displacement. The process of this mutation operator is shown in Figure 5.

- Inversion mutation involves randomly choosing two locations inside a chromosome or tour, then flipping the positions in the substring between these two locations. Figure 6 illustrates the procedure of the inversion mutation operator.

Chromosome	1	2	3	4	5	6	7	8
New Chromosome	1	2	6	7	3	4	5	8

Figure 5: Displacement mutation.

Chromosome	1	2	3	4	5	6	7	8
New Chromosome	1	2	6	4	5	3	7	8

Figure 6: Inversion mutation.

Step 7: Combining

During this step, the created offspring and the existing (parent) population are combined to form a single, bigger population. The multi-objective nature of the problem is then addressed by sorting this combined population according to crowding distance (CD)and Pareto dominance. This preserves high-quality solutions, maintains population variety, and ensures an efficient balance between search space exploration and exploitation by choosing the best subset of individuals to form the next generation.

Step 8: The termination criterion

When the maximum number of generations G_{max} is reached or the population's individuals converge, the suggested algorithm stops. If the termination criteria are met, the non-dominated solutions are offered to the DM; otherwise, proceed to Step 4. The pseudocode of the proposed GA-HM for solving MORAPs is presented in Algorithm 1.

Algorithm 1 : Genetic Algorithm with Hybrid Mutation (GA-HM) for MORAP

Input: Population size N_{pop}, crossover probability P_c, mutation probability P_m, max generations G_{max}
Output: Set of non-dominated (Pareto-optimal) solutions
Initialize population $P(0)$ with N_{pop} random chromosomes (paths)
Evaluate objective functions for each chromosome in $P(0)$
Perform non-dominated sorting to classify solutions into Pareto fronts
for $g = 1$ to G_{max} **do**
    Select randomly N_s parents from $P(g)$ using tournament selection
    Apply uniform crossover with probability P_c to generate $N_c = [P_c \times N_s]$ offspring
    Determine the number of offspring to be mutated as $N_m = [P_m \times N_c]$
    for each selected offspring, generate a random number $r \in [0, 1]$ **do**
        if $r > 0.5$ **then**
            Apply **Displacement Mutation**
        else
            Apply **Inversion Mutation**
        end if
    end for
    Combine parent and offspring populations
    Perform non-dominated sorting and update Pareto fronts
    Select best N_{pop} individuals for next generation
    if termination condition met (max generations G_{max} or convergence) **then**
        Break
    end if
end for
return Final non-dominated Pareto-optimal set

5. Experimental Results

In this section, we apply GA-HM to test problems derived from the literature that involve multi-objective resource allocation [23, 32]. In addition, an engineering application [19] is given to prove that the suggested method can solve engineering difficulties and to confirm that GA-HM works for MORAPs. Intel® Core ™i5 CPU M430 @ 2.27GHz processor WITH 6.00 GB installed memory (RAM) has been used to run all tests. The MATLAB programming language is used to code GA-HM. The GA-HM parameters are set as follows: the maximum number of generations is 100, the crossover rate is 0.1, the mutation rate is 0.02 and the population size is 100.

- Test problem 1 [32]: Allocating six workers to four distinct tasks is the first test problem. You may find the anticipated cost and efficiency in Table 2.

Table 2: Test problem 1's expected cost and efficiency

Number of Workers	Task							
	1		2		3		4	
	Cost	Eff.	Cost	Eff.	Cost	Eff.	Cost	Eff.
00	70.00	0.00	90.00	0.00	85.00	0.00	130	0
01	60.00	25.00	60.00	20.00	60.00	33.00	115	13
02	50.00	42.00	50.00	38.00	50.00	43.00	100	24
03	40.00	55.00	40.00	54.00	55.00	47.00	100	32
04	40.00	63.00	30.00	65.00	40.00	50.00	90	39
05	45.00	69.00	20.00	73.00	30.00	52.00	80	45
06	50.00	74.00	25.00	80.00	25.00	54.00	80	50

This problem is solved by fuzzy dynamic programming (FDP) [32], fuzzy dynamic optimization algorithm (FDOA) [33], effective GA approach (EGAA) [11], k-means-clustering based GA [34], and the proposed GA-HM algorithm. Table 3 explains a comparison of the best-optimized solution that obtained by these algorithms. While the graphical representation of Table 3's solutions is shown in Figure 7. We can infer from the results that FDP introduced four solutions. There is only one non-dominated solution out of the four which is $[x_1, x_2, x_3, x_4] = [2, 3, 1, 0]$. Three non-dominated solutions were identified by the other algorithms, FDOA, EGAA, k-means-clustering based GA, and the suggested GA-HM technique. Furthermore, the three dominated solutions produced by FDP are dominated by these three non-dominated solutions as shown from Table 3 and Figure 7.

Table 3: Solution obtained by compared algorithms for test problem 1

Method	x_1	x_2	x_3	x_4	Cost	Efficiency	Type of Solution
FDP	2	3	1	0	280	129	ND
	3	0	3	0	315	102	Dominated
	5	0	1	0	325	102	Dominated
	2	4	0	0	295	107	Dominated
FDOA	1	2	1	2	270	120	ND
	2	2	1	1	275	126	ND
	2	3	1	0	280	129	ND
EGAA	1	2	1	2	270	120	ND
	2	2	1	1	275	126	ND
	2	3	1	0	280	129	ND
k-means-GA	1	2	1	2	270	120	ND
	2	2	1	1	275	126	ND
	2	3	1	0	280	129	ND
Proposed GA-HM	1	2	1	2	270	120	ND
	2	2	1	1	275	126	ND
	2	3	1	0	280	129	ND

The results for Test Problem 1 demonstrate that GA-HM consistently identifies three non-dominated solutions, similar to FDOA, EGAA, and k-means-based GA, while FDP falls behind with only one. This indicates that GA-HM not only ensures competitive performance but also achieves a balanced trade-off between cost and efficiency. Importantly, the dominated solutions produced by FDP highlight the limitations of dynamic programming in handling multi-objective trade-offs, while the hybridization in GA-HM enhances exploration. The graphical representation (Figure 7) further illustrates that GA-HM solutions are well-distributed across the Pareto front, ensuring decision-makers have more flexibility depending on priorities (cost minimization vs efficiency maximization).

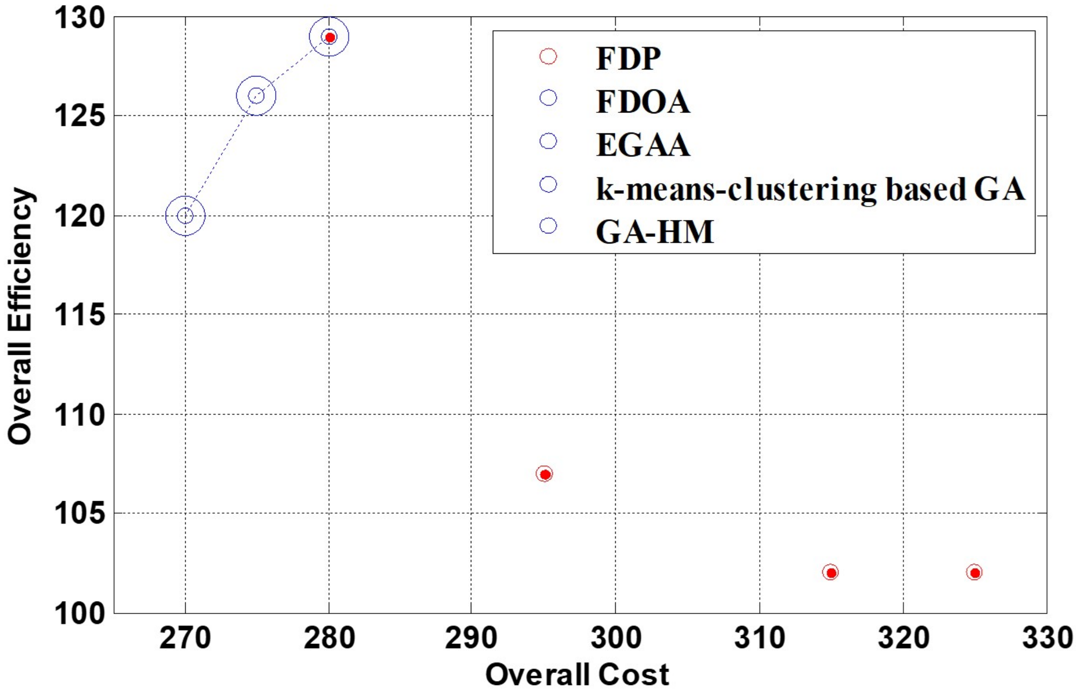

Figure 7: The graphical representation of Table 3's solutions.

- Test problem 2 [23]: Test problem 2 involves assigning ten workers to four specific tasks. The anticipated expense and efficiency are shown in Table 4.

Table 4: Test problem 2's expected cost and efficiency

Number of Workers	Task							
	1		2		3		4	
	Cost	Eff.	Cost	Eff.	Cost	Eff.	Cost	Eff.
00	41.00	0.00	45.00	0.00	36.00	0.00	46.00	0.00
01	38.00	37.00	54.00	49.00	43.00	45.00	78.00	60.00
02	46.00	42.00	36.00	55.00	68.00	49.00	88.00	67.00
03	32.00	50.00	55.00	59.00	56.00	57.00	64.00	72.00
04	78.00	54.00	87.00	62.00	72.00	64.00	90.00	79.00
05	76.00	56.00	82.00	67.00	59.00	77.00	80.00	83.00
06	72.00	58.00	90.00	73.00	32.00	88.00	120.00	88.00
07	84.00	65.00	132.00	80.00	67.00	92.00	104.00	97.00
08	80.00	72.00	97.00	87.00	86.00	100.00	96.00	102.00
09	92.00	80.00	121.00	95.00	88.00	105.00	86.00	110.00
10	96.00	95.00	134.00	102.00	100.00	110.00	120.00	120.00

This problem is solved by multiobjective hybrid genetic algorithm (mo-hGA) [23], k-means-clustering based GA [34], and the proposed GA-HM algorithm. Table 5 explains a comparison of the best-optimized solution that obtained by these algorithms. While the graphical representation of Table 5's solutions is shown in Figure 8.

Table 5: Solution obtained by compared algorithms for test problem 2

Method	x_1	x_2	x_3	x_4	Cost	Eff.	Type of Solution	Type of Solution After Combine
mo-hGA	3	2	1	4	201	229	ND	Dominated
	0	2	6	2	197	210	ND	Dominated
	3	2	5	0	173	182	Dominated	Dominated
	3	1	6	0	164	187	ND	ND
	1	1	6	2	212	241	ND	Dominated
	1	1	5	3	215	235	Dominated	Dominated
	0	1	6	3	191	209	ND	Dominated
k-means-GA	2	1	6	1	210	239	ND	Dominated
	3	1	6	0	164	187	ND	ND
	1	1	6	1	202	234	ND	Dominated
	3	2	2	3	200	226	ND	Dominated
	3	2	1	0	157	150	ND	ND
	3	2	1	3	175	222	ND	ND
	2	2	6	0	160	185	ND	Dominated
	0	2	6	0	155	143	ND	ND
The proposed GA-HM algorithm	1	0	6	3	179	197	ND	Dominated
	0	9	1	0	151	140	ND	ND
	1	9	0	0	141	132	ND	ND
	3	1	6	0	164	187	ND	ND
	2	2	6	0	160	185	ND	ND
	1	2	6	1	184	240	ND	ND
	1	1	6	2	212	241	ND	ND

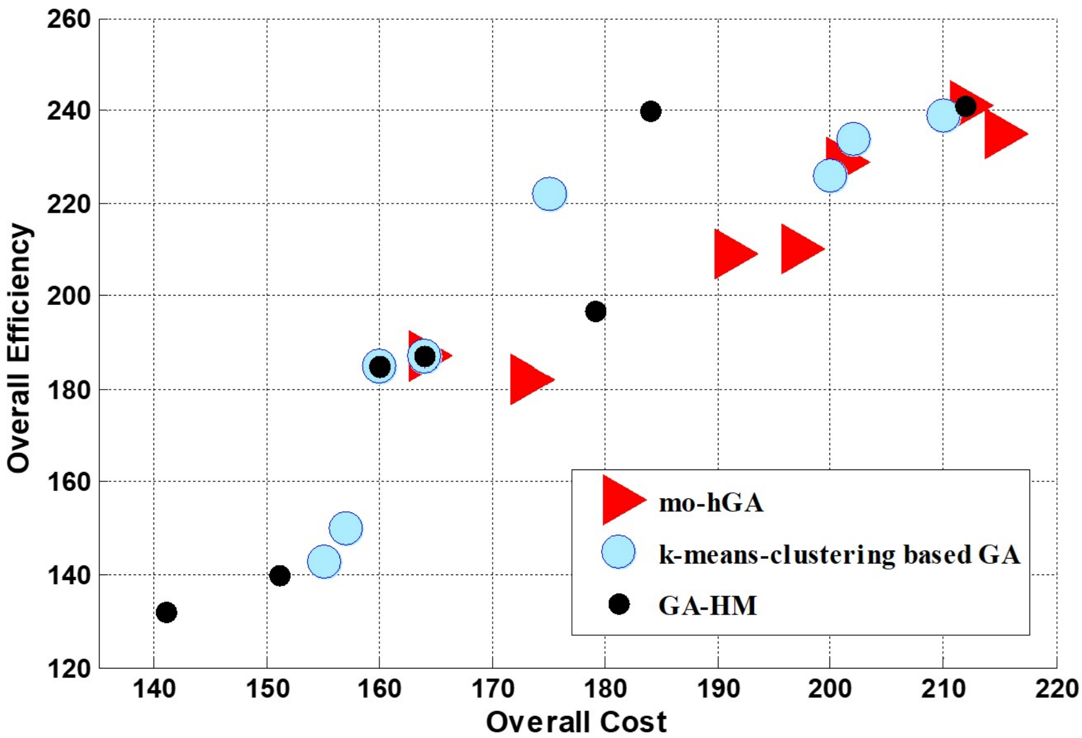

Figure 8: The graphical representation of Table 5's solutions.

Only five of the seven solutions that mo-hGA was able to obtain are non-dominated, according to the results. In contrast, the suggested GA-HM method produced seven non-dominated solutions while the k-means-clustering based GA produced eight non-dominated solutions. But, when we aggregate all of the solutions from the three algorithms, we see that the GA-HM approach has six non-dominated solutions, whilst the k-means-clustering based GA has four, and the mo-hGA has just one.

In Test Problem 2, GA-HM significantly outperforms mo-hGA by producing six non-dominated solutions after combination, compared to only one from mo-hGA. This underlines GA-HM's superior convergence and diversity preservation. While k-means GA generated eight non-dominated solutions initially, only four remained non-dominated after aggregation, suggesting that GA-HM achieves more robust performance. The ability of GA-HM to capture solutions that other methods overlooked demonstrates its strength in exploring the solution space effectively. From an engineering perspective, this implies more reliable and flexible workforce-task allocation under real constraints.

Previous comparisons clearly show that the suggested algorithm's (GA-HM) results are superior to and even dominate those of the other approaches. Also, when compared to the other approaches, GA-HM was able to locate locations that the others missed and had better distribution and spread. Because of this, the suggested approach is in a better position to shed light on the current issues and other multi-objective resource allocation challenges.

This can be explained by the fact that the displacement mutation operator moves the chromosomal segments to new positions within the solution representation. This operation promotes the exploration of different regions of the search space by introducing structural diversity and reducing the risk of premature convergence. In contrast, the inversion mutation operator reverses the order of selected gene segments, generating new solution patterns that can reveal alternative combinations of variables or sequences, potentially leading to higher fitness values.

In summary, incorporating both displacement and inversion mutations into GA-HM allows the algorithm to effectively balance exploration and exploitation. Displacement mutation facilitates broad exploration of the search domain, enabling the algorithm to investigate underrepresented or previously unexplored regions. Meanwhile, inversion mutation enhances local refinement and exploitation of promising areas, thereby improving convergence toward optimal or near-optimal solutions.

6. Computational Complexity

The number of generations, population size, and cost of evaluating the fitness functions are the main factors influencing the computational complexity of the suggested GA-HM algorithm. Let $G_{\max}$ denote the maximum number of generations, N_{pop} the population size, and E_f the time required to evaluate the objective functions for each chromosome. The overall time complexity of GA-HM can therefore be expressed as:

$$O(G_{\max} \times N_{\text{pop}} \times E_f)$$

Compared to the fitness evaluation step, the hybrid mutation's computational overhead is minimal because it just adds two additional operators: displacement and inversion. As a result, the overall asymptotic complexity is still in the same order as that of a standard genetic algorithm.

6.1. Comparison with Other Multi-Objective Genetic Algorithms

Since the additional hybrid mutation procedures only incur a small computational cost, the proposed GA-HM maintains a similar theoretical time complexity of $O(G_{\max} \times N_{\text{pop}} \times E_f)$ when compared to a standard Genetic Algorithm (GA).

The hybrid mutation in GA-HM, which combines displacement and inversion mutations, enhances population diversity and reduces the risk of premature convergence without significantly increasing runtime, in contrast to standard GA or conventional multi-objective algorithms such as NSGA-II. Consequently, GA-HM maintains computational efficiency comparable to well-known GA-based techniques while achieving convergence behavior and a more uniform Pareto front distribution. As shown in Table 6, the proposed GA-HM preserves scalability similar to that of conventional GA, while its hybrid mutation mechanism enables improved Pareto-optimal solutions and more extensive exploration of the search space.

Finally, we conclude that the proposed GA-HM outperforms NSGA-II and K-means GA while maintaining a computational time comparable to a traditional Genetic Algorithm (GA). Its hybrid mutation mechanism, combining displacement and inversion mutations, enhances convergence behavior, increases population diversity, and produces a more uniform and well-distributed Pareto front, all without additional runtime compared to standard GA. As a result, GA-HM provides a better balance between exploration and exploitation of the search space, delivering superior solutions without increasing computational cost. Therefore, GA-HM represents an effective and efficient approach for solving multi-objective resource allocation problems.

Table 6: Comparing the asymptotic time complexity of different algorithms.

Algorithm	**Main Operations**	**Time Complexity**
GA	Selection, crossover, mutation	$O(G_{\max} \times N_{\text{pop}} \times E_f)$
NSGA-II	Non-dominated sorting, CD	$O(Q \times N_{\text{pop}}^2)$
k-means GA	Clustering (k, d) + GA evolution	$O(G_{\max} \times (N_{\text{pop}} \times E_f + k \times d))$
GA-HM	Selection, crossover, hybrid mutation	$O(G_{\max} \times N_{\text{pop}} \times E_f)$

7. Conclusion

This paper proposed the Genetic Algorithm with Hybrid Mutation (GA-HM) as an effective approach to solving the multi-objective resource allocation problem (MORAP). The key contribution of GA-HM lies in its hybrid mutation mechanism, which combines displacement and inversion operators to maintain genetic diversity within the population. By preventing the chromosome population from becoming overly homogeneous, the algorithm successfully avoids premature convergence and entrapment in local minima.

The effectiveness of the proposed method was validated using two benchmark test problems of MORAP. Simulation results demonstrated that GA-HM consistently outperformed other well-established techniques reported in the literature, achieving superior trade-offs between cost and efficiency. Notably, GA-HM was able to generate a diverse set of Pareto-optimal solutions, thereby offering decision-makers greater flexibility in selecting solutions according to specific preferences and priorities.

Based on these findings, it can be concluded that GA-HM is both robust and efficient in addressing MORAPs. Beyond the benchmark cases, the hybrid mutation framework also shows strong potential for application to broader classes of engineering and management optimization problems where maintaining diversity and avoiding local optima are critical for solution quality.

Key directions of MORAP for future research include several promising areas. One important direction is the development of hybrid optimization methods, which integrate metaheuristics with exact techniques to achieve a balance between computational efficiency and solution quality. Another significant area involves dynamic and real-time allocation, focusing on designing adaptive frameworks capable of responding effectively to real-time variations in resource demand and availability. The integration of machine learning is also gaining attention, as predictive models can be employed to anticipate resource requirements and guide optimization processes, thereby enhancing both accuracy and efficiency. Furthermore, addressing robustness under uncertainty is essential and can be achieved through stochastic and robust optimization approaches to handle variability in system parameters, supply, and demand. Additional research should also emphasize the development of decision support systems, creating interactive tools that enable decision-makers to analyze trade-offs among competing objectives and select suitable strategies. Finally, benchmarking and standardization remain critical to the progress of the field by establishing common datasets, problem instances, and evaluation metrics to ensure consistent and transparent comparisons of algorithms.

To promote more effective, resilient, and sustainable resource allocation across various sectors, these directions will strengthen MORAP's theoretical foundations as well as its practical applications.

Finally, future studies should investigate the computational scalability of GA-HM for larger MORAP instances, examining its convergence behavior and efficiency in high-dimensional optimization scenarios. GA-HM can manage more variables and bigger datasets, including scenarios with more employees, activities, or goals. However, factors such as runtime, memory use, and overall speed may degrade as the problem size increases.

Acknowledgements

The authors extend their appreciation to Prince Sattam bin Abdulaziz University for funding this research work through the project number (PSAU/2025/01/34000).

Conflicts of Interest

The authors declare no conflict of interest.

References

[1] J. L. Ponz-Tienda, V. Yepes, E. Pellicer, and J. Moreno-Flores. The resource leveling problem with multiple resources using an adaptive genetic algorithm. *Automation in Construction*, 29:161–172, 2013.

[2] A. Azaron et al. A multi-objective resource allocation problem in pert networks. *European Journal of Operational Research*, 172(3):838–854, 2006.

[3] S. Gupta and T. J. Steenburgh. *Allocating Marketing Resources.* Copyright © 2008, 2008.

[4] X. Wang and G. B. Giannakis. Resource allocation for wireless multiuser ofdm networks. *IEEE Transactions on Communications*, 57(7):4359–4372, 2011.

[5] S. Tan, W. Weng, and S. Fujimura. Scheduling of worker allocation in the manual

labor environment with genetic algorithm. In *International Multi Conference of Engineers and Computer Scientists (IMECS I)*, 2009.

[6] M. Ehrgott, K. Klamroth, and C. Schwehm. An mcdm approach to portfolio optimization. *European Journal of Operational Research*, 155(3):752–770, 2004.

[7] K. Claxton, M. Paulden, H. Gravelle, W. Brouwer, and A. J. Culyer. Discounting and decision making in the economic evaluation of health-care technologies. *Health Economics*, 20(1):2–15, 2011.

[8] T. Ibaraki and N. Katoh. *Resource Allocation Problems: Algorithmic Approaches.* MIT Press, Boston, 1988.

[9] B. Gavish and H. Pirkul. Algorithms for the multi-resource generalized assignment problem. *Management Science*, 37(6):695–713, 1991.

[10] O. Moselhi and P. Lorterapong. Least impact algorithm for resource allocation. *Canadian Journal of Civil Engineering*, 20(2):180–188, 1993.

[11] M. S. Osman, M. A. Abo-Sinna, and A. A. Mousa. An effective genetic algorithm approach to multiobjective resource allocation problems. *Applied Mathematics and Computation*, 163:755–768, 2005.

[12] M. Mutingi. System reliability optimization: A fuzzy multi-objective genetic algorithm approach. *Eksploatacja i Niezawodność – Maintenance and Reliability*, 16(3):400–406, 2014.

[13] D. Bertsimas, J. D. Griffith, V. Gupta, M. J. Kochenderfer, V. V. Mišić, and R. Moss. A comparison of monte carlo tree search and mathematical optimization for large scale dynamic resource allocation. *Preprint submitted to Elsevier*, 2014.

[14] S. K. Chaharsooghi and A. H. M. Kermani. An effective ant colony optimization algorithm (aco) for multi-objective resource allocation problem (morap). *Applied Mathematics and Computation*, 200:167–177, 2008.

[15] Y. C. Liang and C. Y. Chuang. Variable neighborhood search for multi-objective resource allocation problems. *Robotics and Computer-Integrated Manufacturing*, 29(3):73–78, 2013.

[16] J. Wu, Z. Chang, L. Yuan, Y. Hou, and M. Gong. A memetic algorithm for resource allocation problem based on node-weighted graphs. *IEEE Computational Intelligence Magazine*, 2014.

[17] Q. Jia and Y. Seo. An improved particle swarm optimization for the resource-constrained project scheduling problem. *International Journal of Advanced Manufacturing Technology*, 67:2627–2638, 2013.

[18] J. H. Holland. *Adaptation in natural and artificial systems: an introductory analysis with applications to biology, control, and artificial intelligence.* MIT Press, 1992.

[19] D. E. Goldberg. *Genetic Algorithms in Search, Optimization, and Machine Learning.* Addison-Wesley, Reading, MA, 1989.

[20] K. Miettinen. *Nonlinear Multi-objective Optimization.* Kluwer Academic Publishers, Boston, 1999.

[21] A. M. Algelany and M. A. El-Shorbagy. Chaotic enhanced genetic algorithm for solving the nonlinear system of equations. *Computational Intelligence and Neuro-*

science, 2022.

[22] A. Y. Ayoub, M. A. El-Shorbagy, I. M. El-Desoky, and A. A. Mousa. Cell blood image segmentation based on genetic algorithm. In *The International Conference on Artificial Intelligence and Computer Vision*, pages 564–573, Cham, March 2020. Springer International Publishing.

[23] C. M. Lin and M. Gen. Multiobjective resource allocation problem by multistage decision-based hybrid genetic algorithm. *Applied Mathematics and Computation*, 187:574–583, 2007.

[24] N. M. Razali and J. Geraghty. Genetic algorithm performance with different selection strategies in solving tsp. In *Proceedings of the World Congress on Engineering II*, 2011.

[25] C. M. Lin and M. Gen. Multi-criteria human resource allocation for solving multistage combinatorial optimization problems using multi-objective hybrid genetic algorithm. *Expert Systems with Applications*, 34:2480–2490, 2008.

[26] N. Srinivas and K. Deb. Multi-objective optimization using non-dominated sorting in genetic algorithms. *Evolutionary Computation*, 2(3):221–248, 1999.

[27] A. Gelbukh and C. A. Reyes-Garcia. *MICAI 2006: Advances in Artificial Intelligence*. Springer, Heidelberg, 2006.

[28] G. H. Tzeng and J. J. Huang. *Fuzzy multiple objective decision making*. CRC Press, 2013.

[29] M. Gen and R. Cheng. *Genetic Algorithms and Engineering Optimization*. John Wiley & Sons, New York, 2000.

[30] K. J. Wang and Y. S. Lin. Resource allocation by genetic algorithm with fuzzy inference. *Expert Systems with Applications*, 33:1025–1035, 2007.

[31] K. Deep and H. Mebrahtu. Combined mutation operators of genetic algorithm for the travelling salesman problem. *International Journal of Combinatorial Optimization Problems and Informatics*, 2(3):1–23, 2011.

[32] M. L. Hussein and M. A. Abo-Sinna. A fuzzy dynamic approach to the multicriterion resource allocation problem. *Fuzzy Sets and Systems*, 69(2):115–124, 1995.

[33] K. K. Lai and L. Li. A dynamic approach to multiple-objective resource allocation problem. *European Journal of Operational Research*, 117(2):293–309, 1999.

[34] A. A. Mousa, M. A. El-Shorbagy, and M. A. Farag. K-means-clustering based evolutionary algorithm for multi-objective resource allocation problems. *Applied Mathematics and Information Sciences*, 11(6):1681–1692, 2017.