

Article

An Archive-Guided Equilibrium Optimizer Based on Epsilon Dominance for Multi-Objective Optimization Problems

Nour Elhouda Chalabi ¹, Abdelouahab Attia ², Abderraouf Bouziane ², Mahmoud Hassaballah ^{3,4,*},
Abed Alanazi ³ and Adel Binbusayyis ⁵

¹ Computer Science Department, Mohamed Boudiaf University of Msila, Msila 28000, Algeria

² Computer Science Department, University Mohamed El Bachir El Ibrahimi of Bordj Bou Arreridj, Bordj Bou Arreridj 34000, Algeria

³ Department of Computer Science, College of Computer Engineering and Sciences, Prince Sattam Bin Abdulaziz University, AlKharj 16278, Saudi Arabia

⁴ Department of Computer Science, Faculty of Computers and Information, South Valley University, Qena 83523, Egypt

⁵ Department of Software Engineering, College of Computer Engineering and Sciences, Prince Sattam Bin Abdulaziz University, AlKharj 16278, Saudi Arabia

* Correspondence: mah.ali@psau.edu.sa

Abstract: In real-world applications, many problems involve two or more conflicting objectives that need to be optimized at the same time. These are called multi-objective optimization problems (MOPs). To solve these problems, we introduced a guided multi-objective equilibrium optimizer (GMOEO) algorithm based on the equilibrium optimizer (EO), which was inspired by control–volume–mass balance models that use particles (solutions) and their respective concentrations (positions) as search agents in the search space. The GMOEO algorithm involves the integration of an external archive that acts as a guide and stores the optimal Pareto set during the exploration and exploitation of the search space. The key candidate population also acted as a guide, and Pareto dominance was employed to obtain the non-dominated solutions. The principal of ϵ -dominance was employed to update the archive solutions, such that they could then guide the particles to ensure better exploration and diversity during the optimization process. Furthermore, we utilized the fast non-dominated sort (FNS) and crowding distance methods for updating the position of the particles efficiently in order to guarantee fast convergence in the direction of the Pareto optimal set and to maintain diversity. The GMOEO algorithm obtained a set of solutions that achieved the best compromise among the competing objectives. GMOEO was tested and validated against various benchmarks, namely the ZDT and DTLZ test functions. Furthermore, a benchmarking study was conducted using cone- ϵ -dominance as an update strategy for the archive solutions. In addition, several well-known multi-objective algorithms, such as the multi-objective particle-swarm optimization (MOPSO) and the multi-objective grey-wolf optimization (MOGWO), were compared to the proposed algorithm. The experimental results proved definitively that the proposed GMOEO algorithm is a powerful tool for solving MOPs.

Keywords: metaheuristic algorithms; multi-objective optimization; equilibrium optimizer; Pareto solution set; ϵ -dominance relation; cone- ϵ -dominance

MSC: 68T27



check for
updates

Citation: Chalabi, N.E.; Attia, A.; Bouziane, A.; Hassaballah, M.; Alanazi, A.; Binbusayyis, A. An Archive-Guided Equilibrium Optimizer Based on Epsilon Dominance for Multi-Objective Optimization Problems. *Mathematics* **2023**, *11*, 2680. <https://doi.org/10.3390/math11122680>

Academic Editors: Siddhartha Bhattacharya and Mohamed Abd Elaziz

Received: 25 April 2023

Revised: 28 May 2023

Accepted: 5 June 2023

Published: 13 June 2023



Copyright: © 2023 by the authors. Licensee MDPI, Basel, Switzerland. This article is an open access article distributed under the terms and conditions of the Creative Commons Attribution (CC BY) license (<https://creativecommons.org/licenses/by/4.0/>).

1. Introduction

A number of real-life problems have typically been interpreted as optimization problems with multiple conflicting objectives [1,2] (e.g., water distribution networks (WDNs) [3], the traveling salesman problem [4], and protein structures [5]). We are in an era where such problems are increasing daily [6]. In addition, today's decision-making problems

require us to consider large models of these NP-hard problems, both in terms of the number of variables and constraints [7–9]. Therefore, these problems are handled and modeled as multi-objective optimization problems (MOPs), where the goal is to find the best set of trade-off solutions—known as a Pareto optimal set or non-dominated solutions [10]. In other words, this type of optimization searches for acceptable compromises between objectives—as compared to single-objective optimization, which is where only one solution has to be found. Therefore, significant attention has been given to this concept, and many works have been proposed [11,12]. Meta-heuristics and evolutionary algorithms have been widely adopted for solving some of the multi-objective optimization problems, including non-dominating sort genetic algorithms (NSGAII) [13], where fast non-dominated sorting was used. The extension of NSGAII, called NSGAIII [14], employed a non-dominated sort and a reference point method. The PAES [15] and SPEA2 [16] employed an external archive to store the non-dominated solutions; as such, algorithms have been quite successful and are still used today [17,18].

MOPs have been the most common problems in several real-world applications [19,20]. Therefore, this field has continued to evolve, thus ensuring that many other algorithms were developed, such as the multi-objective evolutionary algorithm based on decomposition (MOEAD) [21], where the problem is decomposed into a number of sub-problems and each one is treated as a single-objective problem. Deb et al. [22] introduced the ϵ -MOEA algorithm, where the ϵ -dominance relation was employed. Many other extensions of MOEA have been proposed, including the uniform decomposition measurement (UMOEAD) [23], the MO-memetic algorithm (MOEA/D-SQA) [24], and many others [25,26].

In terms of meta-heuristics algorithms and, particularly, population-based algorithms, algorithms that handle multi-objective problems (MOPs) have typically been an extension of a single-objective-optimization algorithms but modeled in a way to solve MOPs. One of the most well-known algorithms is the multi-objective particle swarm optimization (MOPSO) method, which is based on the single-objective optimization algorithm type of particle swarm optimization (PSO) [27]; it is a population-based algorithm inspired by the biological behavior of birds in a flock. The PSO has been proven to be a successful algorithm that continues to be used for solving optimization problems [28]. Many extended multi-objective versions of the PSO have been proposed. For instance, the swarm metaphor, as proposed in [29], incorporated the Pareto dominance concept and crowding distances. In a different work by Cello et al. [30], another MOPSO was proposed that incorporated a respiratory system to conserve the non-dominated solutions and to choose an instructor that would guide the particles. The well-known algorithm of ant colony optimization (ACO) and its variants [31,32] is another population-based algorithm. It was inspired by ant behavior and designed to solve single-objective optimization problems. Furthermore, it was improved to handle with MOPs accordingly, as in [33–35].

Following the same concept over the years, several other MOP algorithms were developed by simply extending the single-objective version [36,37]. The cat swarm optimization (CSO) [38] method was extended by incorporating a Pareto ranking; thus, it was then named the multi-objective cat-swarm optimization (MOCSSO) [39]. The grey wolf optimizer (GWO) [40], for example, was also extended by adding an external fixed-size archive, resulting in the multi-objective grey wolf optimization (MOGWO) [41] method. Zouache et al. [42] introduced a guided multi-objective moth–flame optimization (MOMFO) method, which was an extension of the moth–flame optimizer (MFO) [43]. In MFO, an unlimited external archive was used to determine the non-dominated solutions, and the fast non-dominated sort was adopted, along with crowding distances. Furthermore, ϵ -dominance was employed as an updated archive strategy. A more recent work attempted to solve MOPs using EO by proposing a multi-objective equilibrium optimizer with an exploration–exploitation dominance strategy (MOEO-EED) [44].

Recently, an equilibrium optimizer algorithm was presented to solve a single-objective optimization problem [45]. The presented results of this algorithm showed that it was able to outperform well-known algorithms. In this paper, based on the aforementioned

analysis and the extended versions, we present an extended version of EO called the guided multi-objective equilibrium optimizer (GMOEO), which we used to solve MOPs. The proposed extension employs an external archive through which to obtain the non-dominated solutions and the crowding distances. In addition, it utilizes an exploration—exploitation dominance, which is where the solutions update was controlled. Furthermore, a Gaussian-based mutation strategy was suggested to boost the exploration and enhance the exploitation. In this work, as compared with MOEO-EED, we attempted to approach the concept using simple strategies.

The main contributions of this paper are summarized as follows.

- We propose a GMOEO method to solve multi-objective optimization problems;
- We incorporated an external archive to store the non-dominated solutions and to guide the particles toward the optimal Pareto set.
- ϵ -dominance was employed to update the archive solutions and to ensure improved diversity, exploitation, and exploration. In addition, cone- ϵ -dominance was employed to update the archive solutions, and was compared with the ϵ -dominance relation;
- A fast non-dominating sort and crowding distances were introduced to preserve the diversity and to ensure the convergence of the particles, as well as to ensure an efficient solution distribution;
- The effectiveness of the proposed algorithm was validated through comprehensive experiments conducted on different benchmarks, including ZDT and DTLZ test functions. The performance was then compared with the known multi-objective optimization algorithms.

The rest of the paper is organized as follows. Section 2 explains the basics of multi-objective optimization problems, Pareto optimality, and EO. Section 3 introduces the proposed GMOEO algorithm. The experimental results, comparisons, and discussion are presented in Section 4. Finally, Section 5 contains the conclusions and suggestions for future research directions.

2. Background Information

2.1. Multi-Objective Optimization Problems

Multi-objective optimization is the systematic process of simultaneously collecting and optimizing conflicting objective functions. The optimization could involve minimization or maximization, depending on the problem. A minimization problem could be formulated as follows:

$$\underset{x \in \mathbb{R}^n}{\text{minimize}} f_i(x), (i = 1, 2, \dots, M), \quad (1)$$

$$\text{subject to } h_j(x) = 0, j = (1, 2, 3, \dots, N), \quad (2)$$

$$g_k(x) \leq 0, k = (1, 2, 3, \dots, L) \quad (3)$$

where $f_i(x)$, $h_j(x)$, and $g_k(x)$ represent the decision functions of a vector $x = (x_1, x_2, x_3, \dots, x_n)^T$, and x_i refers to decision variables. The variables M , $\mathbb{R}^n$, N , and L refer to the objective functions, search space, the numbers of inequality, and equality constraints, respectively. In an MOP, it was difficult to compare the obtained solutions with the relational arithmetic operators. Therefore, the concept of Pareto optimal dominance provided a simple way through which to compare solutions in the multi-objective search space. Furthermore, there was a set of Pareto optimal solutions with a Pareto front image in the search space, rather than a single optimal solution.

2.2. Pareto Dominance and Optimality

The main concept of the Pareto dominance relation comprised the following:

Definition 1 (Pareto dominance). Let z and s be two solutions, where z dominates the other solution s (denote as $z \succ s$), iff:

$$\begin{aligned} \forall i \in \{1, 2, 3, \dots, M\} : f_i(z) \leq f_i(s) \text{ and} \\ \exists j \in \{1, 2, 3, \dots, M\} : f_j(z) < f_j(s) \end{aligned} \quad (4)$$

Solution z weakly dominated the other solution s (denoted as $z \preceq s$), iff:

$$\forall i \in \{1, 2, 3, \dots, M\} : f_i(z) \leq f_i(s) \quad (5)$$

Definition 2 (A non-dominated set). The unique dominant solution where there is no other solution that dominates them. Let D be a set of solutions, such that the non-dominated solutions are members of the set $D' \subseteq D$ and are not dominated by any other solution in set D .

Definition 3. The Pareto solution is an optimal solution if it does not become dominated by other solutions in the feasible search space.

$$\nexists s \in x \mid z \prec s \quad (6)$$

Definition 4 (Pareto optimal set). A set of all non-dominated solutions in the search space.

2.3. Equilibrium Optimizer (EO)

The equilibrium optimization algorithm [45] has recently been introduced to solve optimization problems. The original innovation behind this algorithm was the control-volume-mass balance models, where the mass-balance equation was employed for describing the concentration of a non-reactive constituent in the control volume. It is a first-order differential equation that could provide an understanding of the physics behind the conservation of mass as it enters and leaves the control volume. The solutions represent the particles, while the concentrations represent the positions—similar to PSO. These particles act as a search agent for reaching the optimal solution, which is the equilibrium state. Some of EO's key parameters that distinguish it from other algorithms include the following: an equilibrium pool and a candidate; a generation rate; and updating process for the particles' positions.

The steps of the equilibrium optimizer are summarized in Algorithm 1. Similar to other optimization algorithms, the EO began with the initialization of a random population. As a unique approach, EO employed four of the best-so-far candidates C_{eq1} , C_{eq2} , C_{eq3} , and C_{eq4} as the equilibrium state, which were not initially known. These candidates were necessary to determine the search pattern for the particles, to assist in the exploration of the search space, and to participate in the update process. In addition to the four candidates, an average candidate C_{ave} was calculated. The equilibrium pool was one of the key benefits of this optimizer, and the elements used in the construction of this vector were the five candidates $C_{eq.pool} = \{C_{eq1}, C_{eq2}, C_{eq3}, C_{eq4}, C_{ave}\}$, which were chosen arbitrarily. This pool participated in updating the concentration (position), where a random candidate would be chosen. Other parameters, such as the generation-rate-control parameter GCP and the generation probability GP , were also employed. These two parameters contributed to the construction of the generation rate G , which was another key parameter in the EO algorithm. GCP was used to determine the potential contribution of G during the update process, while GP was used to determine which particle would update its status by using the generation term.

Algorithm 1 Equilibrium optimizer (EO) [45]

```

Initialize the particles  $i = (1, 2, \dots, n)$ 
Assign equilibrium candidates' fitness a large number
Assign the parameters  $a_1, a_2, GP = 0.5$ 
while  $Iteration < Max_{iteration}$  do
  for  $i = 1: \text{number of particles } (n)$  do
    Calculate fitness of  $i^{th}$  particle if  $fitness(C_i) < fitness(C_{eq1})$  then
      | Replace  $C_{eq1}$  with  $C_i$  and  $fitness(C_{eq1})$  with  $fitness(C_i)$ 
    else
      if  $fitness(C_i) > fitness(C_{eq1})$  and  $fitness(C_i) < fitness(C_{eq2})$  then
        | Replace  $C_{eq2}$  with  $C_i$  and  $fitness(C_{eq2})$  with  $fitness(C_i)$ 
      else
        if  $fitness(C_i) > fitness(C_{eq1})$  and  $fitness(C_i) > fitness(C_{eq2})$  and  $(C_i) < fitness(C_{eq3})$  then
          | Replace  $C_{eq3}$  with  $C_i$  and  $fitness(C_{eq3})$  with  $fitness(C_i)$ 
        else
          if  $fitness(C_i) > fitness(C_{eq1})$  and  $fitness(C_i) > fitness(C_{eq2})$  and  $(C_i) > fitness(C_{eq3})$  and  $(C_i) < fitness(C_{eq4})$  then
            | Replace  $C_{eq4}$  with  $C_i$  and  $fitness(C_{eq4})$  with  $fitness(C_i)$ 
          end
        end
      end
    end
  end
   $C_{ave} = (C_{eq1} + C_{eq2} + C_{eq3} + C_{eq4}) / 4$ 
  Equilibrium pool construction  $C_{eq.pool} = C_{eq1}, C_{eq2}, C_{eq3}, C_{eq4}, C_{ave}$ 
  if  $iteration > 1$  then
    | Accomplish memory saving
  end
   $t = (1 - \frac{iteration}{Max_{iteration}})^{(a_2 \frac{iteration}{Max_{iteration}})}$ 
  for  $i = 1: \text{number of particles } (n)$  do
    Randomly choose one candidate from  $C_{eq.pool}$ 
    Randomly generate the values of  $\lambda$  and  $r$  between  $[0, 1]$ 
    Construct  $F = a_1 \text{sign}(r - 0.5)[e^{\lambda t} - 1]$ ,
    Construct  $GCP = \begin{cases} 0.5 & r_1 r_2 \geq GP \\ 0 & r_2 \leq GP \end{cases}$ ,
    Construct  $G_0 = GCP(C_{eq} - \lambda C)$ ,
    Construct  $G = G_0 \cdot F$ ,
    Update Concentrations  $C = C_{eq} + (C_0 - C_{eq}) \cdot F + \frac{G}{\lambda V} (1 - F)$ 
  end
   $Iteration = Iteration + 1$ 
end

```

The generation rate G is one of the most important terms in the EO algorithm for providing the exact solution, which is achieved by improving the exploitation phase. In engineering applications, there are many models that can be used to express the generation rate as a function of time t . For example, one multi-purpose model that describes generation rates as a first-order exponential decay process is defined as

$$\vec{G} = \vec{G}_0 e^{-\vec{k}(t-t_0)} \quad (7)$$

where G_0 is the initial value and k indicates a decay constant. To have a more controlled and systematic search pattern, and to limit the number of random variables, this paper

assumes $k = \lambda$, as well as uses the same previously derived exponential term. Thus, the final set of generation rate is

$$\vec{G} = \vec{G}_0 e^{-\vec{\lambda}(t-t_0)} = \vec{G}_0 \vec{F} \quad (8)$$

The time t is defined as a function of iteration, and is provided in Algorithm 1. More details about the parameters, conditions, and mechanisms used in the EO are provided in [45].

3. The Guided Multi-Objective Equilibrium Optimization

The main concept of the proposed guided multi-objective equilibrium optimization (GMOEO) algorithm was the adoption of an external archive in order to store the discovered non-dominated solutions. The non-dominated solutions were obtained via the Pareto-dominance relation and the crowding distance that improved the diversity and the fast non-dominated sort, which assisted in generating the multiple Pareto fronts. Moreover, the ϵ -dominance relation was introduced for the purpose of updating the archive solutions. Therefore, the best solutions were used to update the archive, and those solutions were from a candidate population that contained the best solutions of the previous population, as well as the current archive solutions. We also used cone- ϵ -dominance to update the archive population solution in order to establish a comparison between these two approaches. Finally, the archive was used to guide the particles in the search space toward the optimal front. To summarize, the adopted strategy in the proposed GMOEO incorporated several important aspects:

- An external archive that could store the best non-dominated solutions in order to guide the particles toward the optimal set;
- The use of an efficient ϵ -dominance/cone- ϵ -dominance relation for updating the archive solutions;
- The integration of a candidate population that enhanced the diversity;
- The use of the fast non-dominated sort (FNS) and the crowding distance to ensure an efficient and diverse set of solutions with an efficient convergence toward the Pareto optimal.

Each of these aspects give the proposed GMOEO a great advantage in being a good tool for optimization, i.e., the archive first helps guide the solution, as well as helps to preserve the diversity of the solutions, which helps with gaining a good set of solutions and also in keeping the balance between exploration and exploitation. The ϵ -dominance relation allows flexibility and diversity, which helps with including a wide range of solutions. The candidate population is a key point for the exploration and exploitation process. Finally, FNS and the crowding distance are the key tools through which to boost the convergence toward the Pareto optimal and coverage.

After the initialization of the population and the evaluation with respect to the M objectives, we first applied the fast non-dominated sort (FNS) on the first population, and then the crowding distance was applied with the aim of recovering the best solutions. Those solutions were the main elements of our candidate population. Next, the GMOEO used an external archive, called the archive population, which was initialized with the non-dominated solution of the first particle population. This archive retained the best solution, or the non-dominated set, during the whole optimization process, including the exploration of the search space. Furthermore, the archive would be used to guide the particles in a later step. To attain the Pareto set solutions during each iteration, the GMOEO had to conduct the following steps: update the candidate population; update the archive population; and, finally, conduct the particle position update.

3.1. Updating the Candidate Population

As previously mentioned, the candidate population was the basis for updating the archive. In other words, this population contributed to the diversity and convergence

toward the Pareto optimal set. In a later step, the EO would use four candidates to manage the movement of the particles. However, in this study, we used the population of the candidates, as this population's candidates were the non-dominated solutions obtained by applying the FNS and the crowding distance. The solutions of this population were guided by a set of two populations during the exploration and exploitation of the search space during the course of the iterations, resulting in a double population. We combined the previous particle population $P(t-1)$ and the current archive population $Arch(t)$ into one population; therefore, these two populations highly contributed to the diversity and the convergence. After combining these two populations, the FNS was considered to preserve the best solutions resulting from both populations. Then, the crowding distance was computed on the last level of the non-dominated solutions. The large distances were selected to complete the candidate population. The three key parameters used to update the candidate population were the following:

3.1.1. Double Population

Double population was one of the key parameters in the update process. As previously stated, the candidate population was updated using two other populations, which included the previous particle population $P(t-1)$ to maintain the diversity. In addition, during exploration and exploitation, the current archive population $Arch(t)$ was utilized to ensure the convergence. The combination of these two distinguishing populations highly contributed to the convergence toward the Pareto optimal set. Furthermore, it aided in avoiding a premature convergence or stagnation in the local optimum set.

$$Double_{pop} = P^{t-1} \cup A^t \quad (9)$$

where P^{t-1} is the previous particle population and A^t represents the current archive populations.

3.1.2. Sorting

Once the double population was obtained, the improved version of FNS [13] was used to compare each solution individually with the rest, rather than storing the results, to avoid duplicate comparisons between the solutions. Then, it sorted the solutions according to the rank of the non-dominated solutions. Furthermore, FNS was employed to sort and maintain the convergence of the double-population solutions. First, the FNS employed each solution of the double population to test its dominance against the other solutions, which resulted in the first non-dominated front. Next, to obtain the individuals of the next front, the first-front solutions were excluded, and the process was repeated until all the possible subsequent fronts were found.

Within this context, each solution for the double population had two inputs: the number of solutions n_i , which dominated the i^{th} solution, and a set of solutions S_i , which were dominated by the i^{th} solution. For $n_i = 0$, its solution was assigned to the sub-front $F1$. For all the solutions of the current front $F1$, every solution (j) of the S_i set traveled, and its n_j count was decreased. However, if $n_j = 0$, then this solution was set to another sub-set H . After reviewing all of the individuals of the current front, $F1$ was then announced as the first front. This process was repeated in subset H . Lastly, these solutions were saved based on their front, as is described in Algorithm 2.

This mechanism was employed for estimating the density of a certain solution, which preserved the diversity and supported an efficient distributed solution. The crowding distance that was applied on the population was determined by the application of FNS. The crowding distance was computed as the average distances of two neighboring solutions by considering the respective objectives from all sides. Figure 1 shows the crowding distance of a solution i as the average distance of the cuboid, and is represented by the two closest neighbors $i+1$ and $i-1$. The solutions were ranked in ascending order based on their crowding distances in objective m . Then, the boundary solutions with the highest and lowest objective values were set to be infinity. The steps of the crowding distance are

outlined in Algorithm 3. It should be noted that $F[i]_m$ represents the m^{th} objective function associate with the i^{th} solution in front of F . Once the crowding distance values had been obtained, the solutions were then ranked according to the crowding distance values, and the first n solutions were selected as the new candidate population.

Algorithm 2 Steps of the fast non-dominated sort algorithm.

Input: Double population

```

foreach  $x \in \text{Double population}$  do
  foreach  $y \in \text{Double population}$  do
    if  $x$  dominate  $y$  then
      | Add  $y$  to  $S_x$ 
    else
      if  $x$  is dominated by  $y$  then
        | Increase the count  $n_x$ 
      end
    end
  end
  if no solution dominated  $x$  then
     $x$  Belong to the first front  $F$ 
    while  $F_i \neq 0$  do
      foreach individual  $pinF_i$  do
        foreach modified member from the set  $S_x$  do
          | Decrease the count  $n_y$  by one
          if  $n_y = 0$ ,  $y$  member in  $H$  then
            | Current front is formed from the  $H$  set members
          end
        end
      end
    end
  end
end

```

Algorithm 3 Steps to compute crowding distance for each solution.

Input: F population based Front;

N solutions number in the front F

foreach k initialize the distance by Zero **do**

```

  foreach objective  $m$  do
    Rank  $F$  according to objective value
    Assigned infinity  $\infty$  to the boundary solutions
    for  $k = 2$  to  $(k - 1)$  do
      |  $F[k]$  distance =  $F[k]$  distance +  $(F[k + 1]_m$  distance -  $F[k - 1]_m$  distance)
    end
  end
end

```

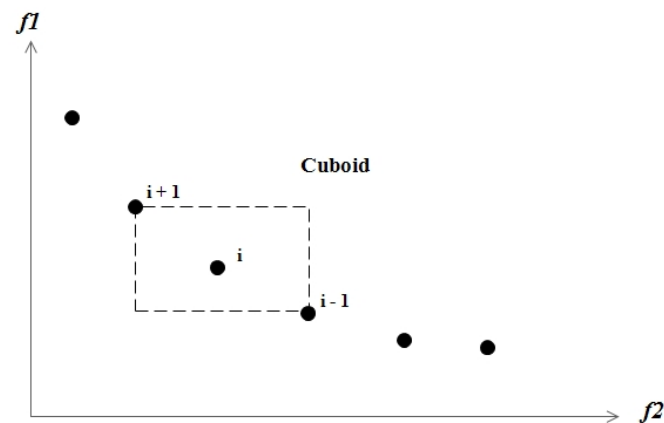


Figure 1. Crowding distance calculation.

3.2. Updating the Archive Population

In order to ensure the performance of the proposed GMOEO, an external archive was used. This archive assisted in maintaining the non-dominated solution and cooperated in the movement of the particles toward the Pareto front. In other words, this archive acted as the equilibrium pool. Two benchmarking strategies were adapted for updating the archive's solutions: first, we considered the ϵ -dominance as an update strategy; second, we employed the cone- ϵ -dominance.

3.2.1. ϵ -Dominance

The ϵ -dominance [46] basically adapted two main mechanisms: box-level and regular dominance. It began as ϵ -dominance splits the objective space into hyper-boxes, where each box contained one unique identification vector B for every solution from the archive population $Arch(t)$. Then, the candidate population $Candidate_{pop}(t)$ —where $B = (B_1, B_2, \dots, B_M)$ for the M objectives—was determined by the following:

$$B_i(f) = \lfloor \frac{\log(f_i)}{\log(\epsilon + 1)} \rfloor \quad (10)$$

where f_i is the objective value of the i^{th} solution and ϵ refers to the admissible error. Once the identification vectors values were computed, each candidate population solution was then compared to all the archive solutions, which was based on the ϵ -dominance relation, to determine whether to add the solution to the archive. The box-level dominance was conducted to ensure the diversity. If the identification vector of the candidate population solution, denoted as B_c , dominated an identification vector of the archive solution, denoted as BA_i , then c would be stored in the archive and the archive solution A_i would be removed. Otherwise (i.e., if B_c was dominated by a BA_i), the c solution would not be stored. If neither result occurred, a second mechanism was employed, i.e., a regular dominance $BA_i = B_c$. If c dominated A_i , then c would be accepted. If there was no clearly dominant solution, then the closest solution to B would be accepted. Algorithm 4 describes the procedure for updating the archive using ϵ -dominance.

The main reason behind the choice of ϵ -dominance is first Pareto optimality, which is the ϵ -dominance relation that can provide a Pareto optimal set in a conflicting objectives space. ϵ -dominance uses the parameter ϵ , which allows for tolerance and flexibility while choosing the optimal solutions that are reflected in the diversity of the solutions. By setting ϵ , there is possibility to obtain different ranks for the solutions in the optimal set. Moreover, ϵ -dominance is scalable, and it can handle problems with a large number of objectives. Lastly, this relation is computationally efficient since the comparison involves only two solutions at a time.

Algorithm 4 Archive updating using ϵ -dominance.

Input: Archive population $A(t)$, iteration number t , candidate population c solution.
 Calculate vector B_c and BA for all archive population solutions $A(t)$,
if $\exists x \in A(t) | B_x \succ B_c$ **then**
 | c is rejected
end
if $\exists x \in A(t) | B_c \succ B_x$ **then**
 | c replace $x \in A(t)$
end
if both above cases does not occurs then then
 if $\exists x \in A(t) | B_c \sim B_x$ **then**
 if $c \sim x$ **then**
 | Keep the solution with the smallest distance to vector B
 else
 | Retain the solution dominating all other solutions
 end
 else
 | Add the solution c to the archive $A(t)$
 end
end

3.2.2. Cone- ϵ -Dominance

Cone- ϵ -dominance [47] is a relaxed version of Pareto dominance, which itself is a relaxed dominance approach. Relaxed dominance [48] was introduced to handle situations such as when a solution has a significantly inferior value to one of the objectives. However, it was not dominated in this study. The α -dominance was adopted, and to select the non-dominated solutions with more flexibility, it used linear trade-off functions and set an upper and lower rate between the two distinguishing objectives. Then, it could permit solutions to dominate each other with a significant superiority in only one objective. Consequently, a significant superiority in another objective would typically be rejected by a regular Pareto dominance. Based on the relaxed dominance, cone- ϵ -dominance was a hybridization of both the α -dominance and the ϵ -dominance. It attempted to retain the convergence of ϵ -dominance while controlling the dominant region using cones.

Definition 5 (Cone). A set H was indicated as a cone Cone if $\lambda x \in H$ for any $x \in H \forall \lambda \geq 0$.

Definition 6 (Generated Cone). For two vectors denoted by y_1 and y_2 , the cone generated by these two vectors is a set H described by the following:

$$H = \{z : z = \lambda_1 y_1 + \lambda_2 y_2, \forall \lambda_1, \lambda_2 \geq 0\} \quad (11)$$

For m dimension, y_i , and $i \in \{1, 2, \dots, m\}$, the set H becomes

$$H = \{z : z = \lambda_1 y_1 + \dots + \lambda_m y_m + \dots + \lambda_i y_i, \forall \lambda_i \geq 0\} \quad (12)$$

With respect to the origin of the box $y_1 = [\epsilon_1 k \epsilon_2]^T$ and $y_2 = [\epsilon_1 k \epsilon_2]^T$, the cone would then be determined, as follows:

$$H = \left\{ z : \begin{bmatrix} z \\ z_1 \\ z_2 \end{bmatrix} = \overbrace{\begin{bmatrix} \epsilon_1 & k \epsilon_2 \\ k \epsilon_1 & \epsilon_2 \end{bmatrix}}^{\psi} \overbrace{\begin{bmatrix} \lambda_1 \\ \lambda_2 \end{bmatrix}}^{\lambda}, \forall \lambda_1, \lambda_2 \geq 0 \right\} \quad (13)$$

where $k \in [0, 1)$ is a parameter to control the cone opening and ψ is the cone-dominance matrix.

Similar to ϵ -dominance, the cone- ϵ -dominance splits the objective spaces into hyper-boxes. In addition, it adapts two levels of dominance: regular Pareto and box level. The box-level dominance contained one unique identification vector b with upper and lower bounds, such as the property adapted by α -dominance. Every solution from the $Arch(t)$ archive population and the $Candidate_{pop}(t)$ candidate population was assigned a box where b would be calculated using the following:

$$b_i = \begin{cases} \epsilon_i \lfloor \frac{x_i}{\epsilon_i} \rfloor, i \in \{1, 2, \dots, m\} & \text{for minimizing } f_i \\ \epsilon_i \lceil \frac{x_i}{\epsilon_i} \rceil, i \in \{1, 2, \dots, m\} & \text{for maximizing } f_i \end{cases} \quad (14)$$

where x represents a solution $\lfloor \cdot \rfloor$, which returns the closest low integer to their argument, and $\lceil \cdot \rceil$ returns the closest high integer to their argument.

To add a solution to the archive, a box-level dominance was conducted. Let b_c and b_a be the two identification vectors of the candidate population solution and the archive population solution, respectively. Furthermore, b_c was the cone- ϵ -dominate b_a if, and only if, b_c was the Pareto dominate b_a ; otherwise, the solution to the linear system was $\psi\lambda = z$, as determined by the following:

$$z = b_a - [b_c - \epsilon], \epsilon_i > 0 \quad (15)$$

such that

$$\lambda_i \geq 0, \forall i \in \{1, \dots, m\} \quad (16)$$

Similarly, $b_c \prec^{cone \epsilon} b_a$ if, and only if:

$$(b_c \prec b_a) \vee (\psi\lambda = z | \lambda_i \geq 0, i \in \{1, \dots, m\}) \quad (17)$$

For the identification vectors with the same box, a regular Pareto dominance was employed, where the archive solution would be replaced only if the candidate solution was a Pareto dominant. Otherwise, the closest point to the origin of the box was selected. Algorithm 5 shows the procedure for updating the archive using the cone- ϵ -dominance.

The cone- ϵ -dominance relation is as efficient as the ϵ -dominance relation; however, it offers more flexibility in defining the region around a solution. In addition cone- ϵ -dominance allows more accurate representation and coverage.

3.2.3. Evaluating the Archive Population Size

During archive updating, the total number of the solutions can be increased, which would then expand the archive. Therefore, to control the number of the solutions contained in the archive population, we utilized the FNS strategy on the archive population. Then, the crowding distance values computed for the solutions were sorted in descending order, and only the first 100 solutions from this were retained.

3.3. Updating the Particle Position

The strategy for updating the particle positions, adapted for the GMOEO algorithm, was the same as in EO, as given in Algorithm 1 for updating the concentrations. The only difference in this step was that, instead of using an equilibrium pool with only five candidates, a pool that contained those five candidate in addition to the archive population was used. As a result, we could guarantee the convergence toward the Pareto optimal since the archive contained non-dominated solutions. Algorithm 6 describes the process for updating the particle positions. Using GMOEO for each particle, a solution was randomly selected from the equilibrium pool, i.e., the archive population and the five candidates, and was used in the update process.

Algorithm 5 Archive updating using the cone ϵ -dominance.

Input: Archive population A , $c \in$ candidate population solution
 Calculate vector b_c and b for all archive population solutions A
if c is cone- ϵ -dominated by any $a \in A$ **then**
 Reject c
else
 if c share same box as a **then**
 if c dominates a or near to box origin a **then**
 Remove all cone- ϵ -dominated archive
 Exchange a by c
 else
 Reject c
 end
 else
 if c cone- ϵ -dominates any $a \in A$ **then**
 Remove all cone- ϵ -dominated archive
 Add c into the archive
 else
 Add c into the archive
 end
 end
end

Algorithm 6 Updating particles position.

Input: Particles population P ,
 Archive population A , $C_{eq1}, C_{eq2}, C_{eq3}, C_{eq4}$,
 $C_{ave} = (C_{eq1} + C_{eq2} + C_{eq3} + C_{eq4}) / 4$,
 Equilibrium pool construction:
 $C_{eq.pool} = C_{eq1}, C_{eq2}, C_{eq3}, C_{eq4}, C_{ave}, A$
for $i = 1$: number of particles (n) **do**
 Randomly choose one candidate from $C_{eq.pool}$,
 Randomly generate the values of λ and r in $[0, 1]$,
 Construct $F = a_1 \text{sign}(r - 0.5)[e^{\lambda t} - 1]$,
 Construct $GCP = \begin{cases} 0.5 & r_1 r_2 \geq GP \\ 0 & r_2 \leq GP \end{cases}$,
 Construct $G_0 = GCP(C_{eq} - \lambda C)$,
 Construct $G = G_0 \cdot F$,
 Update concentrations $C = C_{eq} + (C_0 - C_{eq}) \cdot F + \frac{G}{\lambda V}(1 - F)$
end

For the computational complexity of GMOEO, we let N be the size of the population and M the number of objectives. The main loop of the algorithm had the same complexity as the EO [45]. It used a polynomial order of $O(tdN + tcN)$, where t is the number of iteration, c is the cost function, and d is the problem dimension. The first operation was FNS, which had a complexity of $O(MN^2)$. The second operation was the crowding distance of complexity $O(2N \log(2N))$. Finally, updating the archive required the use of ϵ -dominance, where the complexity of the operation was $O(MN^2)$, and for the cone- ϵ -dominance, the complexity was $O(M^2N)$. Consequently, the processing time of the GMOEO algorithm was $\max [O(tdN + tcN), O(MN^2), O(2N \log(2N)), O(MN^2), O(M^2N)]$. However, the complexity of GMOEO was $O(MN^2)$, which showed the same complexity as MOPSO [30] and MOGWO [41]. Algorithm 7 illustrates all the steps of the GMOEO with the additional operations. In addition, Figure 2 displays the flowchart for the proposed GMOEO.

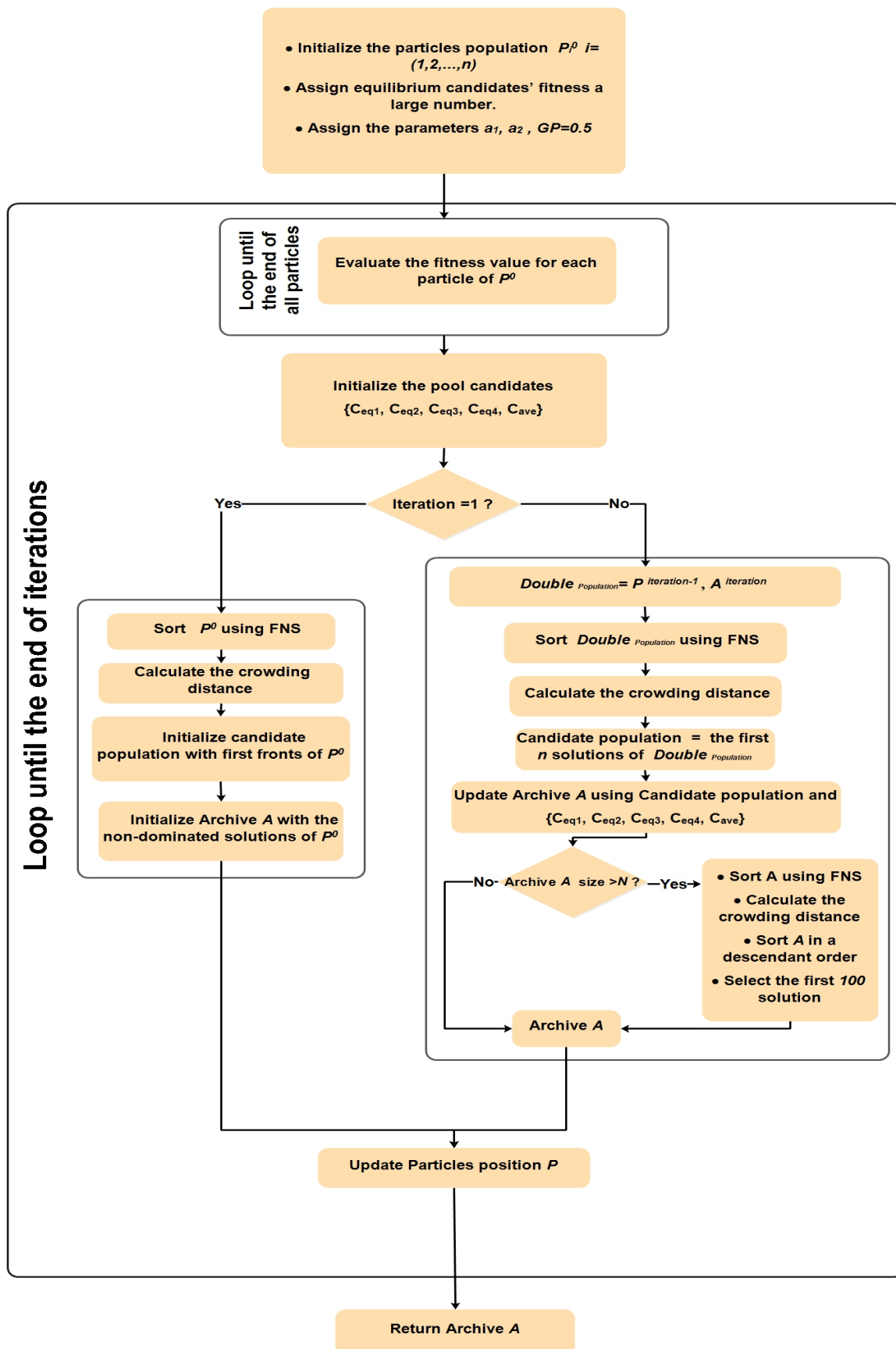


Figure 2. Flowchart of the proposed GMOEO algorithm.

Algorithm 7 The guided multi-objective equilibrium optimizer (GMOEO).

```

Initialize the particles  $i = (1, 2, \dots, n)$ ,
Assign equilibrium candidates' fitness a large number,
Assign the parameters  $a_1, a_2, GP = 0.5$ 
while  $Iteration < Max_{iteration}$  do
  for  $i = 1$ : number of particles ( $n$ ) do
    Calculate fitness of  $i^{th}$  particle,
    Initialize  $C_{eq1}, C_{eq2}, C_{eq3}, C_{eq4}$  same as in EO
  end
  if  $iteration = 1$  then
    Sort  $P^0$  using FNS
    Compute the crowding distance for non-dominated solutions,
    Initialize the candidate population with first fronts of the particle  $P^0$ ,
    Initialize the archive  $A$  with the non-dominated solutions of the particle population  $P^0$ 
  else
     $Double_{population} = \text{Combine} [\text{current archive population } A^{iteration}, \text{previous particleal population } P^{iteration-1}]$ 
    Sort  $Double_{population}$  using FNS
    Compute the crowding distance for non-dominated solutions
    Candidate population = first  $n$  solutions
    Update archive population  $A$  using candidate population and  $\{C_{eq1}, C_{eq2}, C_{eq3}, C_{eq4}\}$ 
    if  $archive\ population\ A_{size} > 100$  then
      Sort  $A$  using FNS
      Compute crowding distance
      Sort in descendent order based on the crowding distances
      Select the first 100 solutions
    end
    Update particles position  $P$ 
     $iteration = iteration + 1$ 
  end
end
Return: Archive population  $A$ 

```

4. Experimental Results

For evaluating the proposed GMOEO algorithm, several experiments were conducted with 12 different benchmarks, including the test functions of the ZDT series [49] and the DTLZ series [50]. The benchmarking test functions and their properties are reported in Table 1. To further validate our findings, the proposed GMOEO was compared, both qualitatively and quantitatively, with several well-known multi-objective optimization algorithms: namely, the guided multi-objective equilibrium optimizer (with cone- ϵ -dominance) [45], the multi-objective particle-swarm optimization (MOPSO) [30] method, and the multi-objective grey wolf optimization (MOGWO) [41] method. In all experiments, the number of operations was set at 10, the number of iterations was set at 6000, and the population size was equal to 40. These settings were the same for all algorithms and functions. In our discussion of the results, we refer to GMOEO using an ϵ -dominance with ϵ -GMOEO, and also to GMOEO using cone- ϵ -dominance with cone- ϵ -GMOEO.

The comparison was conducted based on three metrics: spacing (SP) [51], maximum spread (MS) [52], and inverted generational distance (IGD) [53].

Table 1. Characteristics of the multi-objective test functions.

Bi-objective test functions	
Test function	Characteristics
ZDT1	with a convex front
ZDT2	with a non-convex front
ZDT3	with a discontinuous front
ZDT4	with 2^{21} local Pareto-optimal fronts and therefore is highly multi-modal
ZDT6	with a non-uniform search space
Three-objective test functions	
Test function	Characteristics
DTLZ1	with a linear Pareto-optimal front
DTLZ2	with a spherical Pareto-optimal front
DTLZ3	with a many Pareto-optimal fronts
DTLZ4	with Pareto-optimal front has dense set of solutions to exist near the $f_M - f_1$
DTLZ5	this problem will verify the ability of MOEA to converge to a degenerated curve
DTLZ6	this problem has 2^{M-1} disconnected Pareto-optimal front.
DTLZ7	this problem has Pareto-optimal front which is a combination of a straight line and a hyper-plane.

- **SP** was used to evaluate the uniformity of the distribution of non-dominated solutions and the diversity of the solutions. The spacing assisted in estimating the distribution of the obtained solution along the Pareto front. SP was defined as follows:

$$SP = \sqrt{\frac{1}{n-1} \sum_{i=1}^n (\bar{d} - d_i)^2} \quad (18)$$

where

$$d_i = \min_j \left(\sum_{k=1}^m |f_m^i - f_m^j| \right) \quad (19)$$

where $i, j = 1, 2, \dots, n$, with n as the number of solutions obtained in the front, while $\bar{d}$ is the average of all d_i with m as the number of objective functions f ;

- **MS** was used to measure the diagonal length of the hyper-box generated by the extreme values of the objective functions in the non-dominated solution set. MS was defined as follows:

$$MS = \sqrt{\sum_{i=1}^M \max(d(a_i, b_i))} \quad (20)$$

where d represents the Euclidean distance between a_i and the maximum value of the i th objective function, and b_i is the minimum value of the i th objective function with M as the number of the objective functions;

- **IGD** was an inversion of the generational distance metric, which could be computed as the distance between the reference Pareto front and each of the closest non-dominated

solutions. Generally, IGD is used as a measure of an algorithm's convergence. It was formulated as follows:

$$IGD = \frac{\sqrt{\sum_{i=1}^n d_i^2}}{n} \quad (21)$$

where n represents the number of the true Pareto solution set, while d_i^2 represents the Euclidean distance between the nearest Pareto solution obtained by the algorithm and the true Pareto front: the smaller the IGD and SP, the better the performance. Therefore, these were used to perform the comparison accordingly.

4.1. Results of the ZDT Test Functions

Tables 2–4 summarize the statistical results of the metrics IGD, SP, and MS for the ZDT test functions, respectively. Furthermore, to confirm our findings, the qualitative significations of the results are illustrated in Figures 3–7. Table 2 lists the statistical results of the IGD metric for the proposed ϵ -GMOEO, cone- ϵ -GMOEO, MOPSO, and MOGWO methods, along with the values for best, worst, average, median, and standard deviations. Overall, in 10 independent operations, the proposed ϵ -GMOEO and cone- ϵ -GMOEO outperformed the well-known MOPSO and MOGWO according to the test functions (ZDT1, ZDT2, ZDT4, and ZDT6). However, ϵ -GMOEO provided better results than cone- ϵ -GMOEO.

Table 2. Results for the IGD of the ZDT-series test functions.

Algorithm	Best	Worst	Average	Median	Std
ZDT1					
ϵ -GMOEO	1.457.e-04	1.537.e-04	1.484.e-04	1.481.e-04	2.67.e-06
cone- ϵ -GMOEO	1.938.e-04	1.149.e-03	3.718.e-04	2.424.e-04	2.989.e-04
MOPSO	4.02.e-04	5.885.e-04	4.741.e-04	4.468.e-04	6.61.e-05
MOGWO	3.9.e-04	5.827.e-04	4.702.e-04	4.586.e-04	7.112.e-05
ZDT2					
ϵ -GMOEO	1.4812.e-04	1.557.e-04	1.507.e-04	1.491.e-04	3.048.e-06
cone- ϵ -GMOEO	2.311.e-02	2.312.e-02	2.311.e-02	2.311.e-02	3.468.e-06
MOPSO	5.117.e-04	3.268.e-02	2.947.e-02	3.268.e-02	1.017.e-02
MOGWO	5.034.e-04	3.268.e-02	2.625.e-02	3.268.e-02	1.356.e-02
ZDT3					
ϵ -GMOEO	9.031.e-03	9.093.e-03	9.07.e-03	9.078.e-03	2.263.e-05
cone- ϵ -GMOEO	1.716.e-03	2.321.e-02	6.072.e-03	4.186.e-03	6.442.e-03
MOPSO	1.103.e-02	2.041.e-02	1.484.e-02	1.123.e-02	4.784.e-03
MOGWO	4.550.e-04	6.723.e-04	5.549.e-04	5.380.e-04	6.8.e-05
ZDT4					
ϵ -GMOEO	1.458.e-04	3.501.e-02	6.506.e-03	1.479.e-04	1.348.e-02
cone- ϵ -GMOEO	2.724.e-01	2.011	0.931	0.853	0.47
MOPSO	7.661.e-02	7.387.e-01	3.314.e-01	2.797.e-01	2.277.e-01
MOGWO	5.076.e-04	1.727.e-01	6.208.e-02	4.083.e-02	5.689.e-02
ZDT6					
ϵ -GMOEO	1.177.e-04	1.205.e-04	1.188.e-04	1.186.e-04	8.225.e-07
cone- ϵ -GMOEO	1.552.e-03	3.238.e-02	1.048.e-02	9.128.e-03	8.442.e-03
MOPSO	2.918.e-04	6.732.e-04	3.978.e-04	3.711.e-04	1.040.e-04
MOGWO	3.401.e-04	5.679.e-04	4.691.e-04	4.621.e-04	6.797.e-05

As for ZDT3 test function, the results proved that MOGWO achieved efficient results as it was able to converge toward all the disconnected fronts, while cone- ϵ -GMOEO was

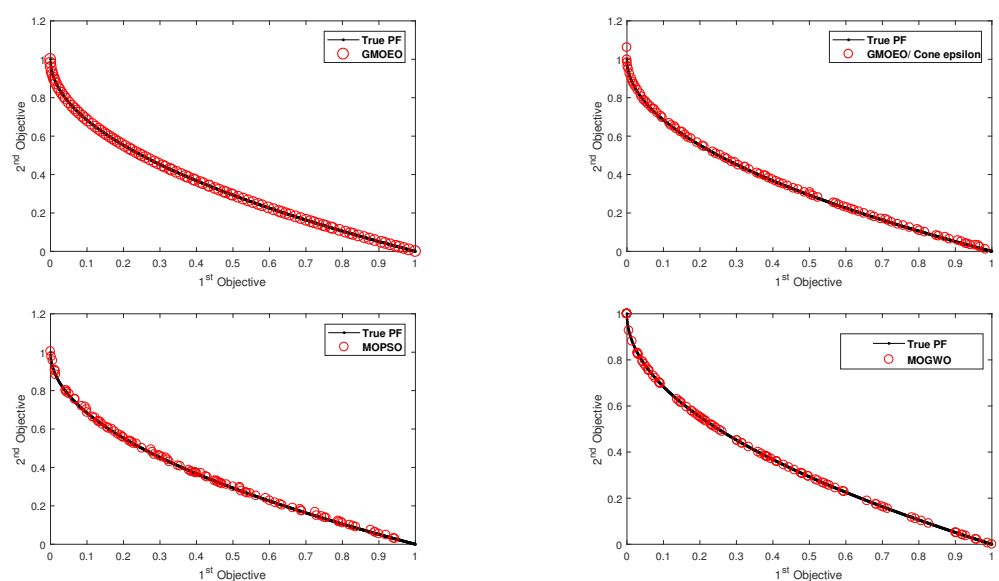
able to outperform both MOPSO and ϵ -GMOEO. Moreover, the reported results related to diversity, which are SP in Table 3 and MS in Table 4, showed that ϵ -GMOEO clearly outperformed the other benchmarking algorithms in the bi-objective test function. In addition, the results in Figures 3–7 for ZDT1, ZDT2, ZDT4, and ZDT6, respectively, clearly supported our statistical findings and showed that the proposed ϵ -GMOEO had better coverage, diversity, and convergence toward the Pareto optimal when compared to the cone- ϵ -GMOEO, MOPSO, and MOGWO methods. Figure 5 represents the test function ZDT3, as shown in the algorithm, and it shows that MOGWO could perfectly converge toward the disconnected fronts, though cone- ϵ -GMOEO was also able to converge toward four fronts. Similar to MOPSO, ϵ -GMOEO was able to converge toward three fronts. However, ϵ -GMOEO showed better coverage. Regarding the IGD value for ZDT3, ϵ -GMOEO ranked second after MOGWO. Table 3 shows the statistical results for the SP metric, which proved that the proposed algorithm was highly competitive as ϵ -GMOEO had the best results in all the ZDT test functions when compared to the other algorithms. The ϵ -GMOEO method had a better distribution over the Pareto front. As shown in Figures 3–7, the front generated by ϵ -GMOEO was well distributed and extended in all the test functions. The cone- ϵ -GMOEO method ranked second and was also able to outperform MOPSO and MOGWO in three of the test functions (ZDT1, ZDT2, and ZDT 6). Table 4 shows the results for the MS metric, indicating that ϵ -GMOEO was also able to provide efficient results. These outcomes clearly confirmed that our proposed ϵ -GMOEO had better diversity in all the test functions. In addition, the cone- ϵ -GMOEO was also able to compete efficiently in most of the test functions when compared to the other two algorithms (i.e., MOPSO and MOGWO).

Table 3. Results for the SP of the ZDT series test functions.

Algorithm	Best	Worst	Average	Median	Std
ZDT1					
ϵ -GMOEO	3.561.e-03	4.489.e-03	3.995.e-03	3.980.e-03	2.427.e-04
cone- ϵ -GMOEO	5.536.e-03	9.948.e-03	6.945.e-03	6.597.e-03	1.319.e-03
MOPSO	5.939.e-03	8.457.e-03	7.748.e-03	8.014.e-03	7.512.e-04
MOGWO	5.910.e-03	8.818.e-03	7.925.e-03	8.041.e-03	8.252.e-04
ZDT2					
ϵ -GMOEO	3.136.e-03	4.343.e-03	3.717.e-03	3.686.e-03	3.741.e-04
cone- ϵ -GMOEO	5.595.e-01	5.6173.e-01	NaN	NaN	NaN
MOPSO	0	5.743.e-03	5.743.e-04	0	1.816.e-03
MOGWO	0	9.298.e-03	1.725.e-03	0	3.650.e-03
ZDT3					
ϵ -GMOEO	2.875.e-03	4.324.e-03	3.562.e-03	3.783.e-03	5.303.e-04
cone- ϵ -GMOEO	1.117.e-02	8.719.e-02	3.704.e-02	2.484.e-02	2.564.e-02
MOPSO	4.023.e-03	6.853.e-03	5.317.e-03	5.302.e-03	8.792.e-04
MOGWO	9.662.e-03	1.776.e-02	1.327.e-02	1.333.e-02	2.539.e-03
ZDT4					
ϵ -GMOEO	1.778.e-03	4.053.e-03	3.429.e-03	3.468.e-03	6.892.e-04
cone- ϵ -GMOEO	7.574.e-03	2.509.e-02	1.558.e-02	1.653.e-02	5.0318.e-03
MOPSO	3.809.e-03	1.093.e-02	6.698.e-03	6.344.e-03	1.885.e-03
MOGWO	0	7.6716.e-03	NaN	NaN	NaN
ZDT6					
ϵ -GMOEO	2.941.e-03	3.188.e-02	5.993.e-03	3.125.e-03	9.096.e-03
cone- ϵ -GMOEO	3.476.e-03	5.731.e-02	1.244.e-02	6.353.e-03	1.639.e-02
MOPSO	3.793.e-03	6.614.e-03	5.404.e-03	5.449.e-03	8.192.e-04
MOGWO	5.935.e-03	8.491.e-03	7.346.e-03	7.336.e-03	9.165.e-04

Table 4. Results for the MS of the ZDT-series test functions.

Algorithm	Best	Worst	Average	Median	Std
ZDT1					
ϵ -GMOEO	1	1	1	1	9.617.e-08
cone- ϵ -GMOEO	9.321.e-01	9.807.e-01	9.670.e-01	9.697.e-01	1.366.e-02
MOPSO	9.832.e-01	9.978.e-01	9.890.e-01	9.876.e-01	5.273.e-03
MOGWO	9.969.e-01	1	9.996.e-01	1	9.677.e-04
ZDT2					
ϵ -GMOEO	1	1	1	1	2.384.e-07
cone- ϵ -GMOEO	7.070.e-01	7.071.e-01	NaN	NaN	NaN
MOPSO	9.842.e-01	9.842.e-01	NaN	NaN	NaN
MOGWO	1	1	NaN	NaN	NaN
ZDT3					
ϵ -GMOEO	1	1	1	1	7.105.e-07
cone- ϵ -GMOEO	7.428.e-01	9.942.e-01	9.622.e-01	9.86.e-01	7.732.e-02
MOPSO	8.855.e-01	1	9.538.e-01	9.981.e-01	5.851.e-02
MOGWO	1	1	1	1	3.285.e-06
ZDT4					
ϵ -GMOEO	7.071.e-01	1	9.428.e-01	1	1.206.e-01
cone- ϵ -GMOEO	7.23.e-01	8.625.e-01	7.812.e-01	7.728.e-01	5.347.e-02
MOPSO	7.297.e-01	8.791.e-01	8.112.e-01	8.158.e-01	5.253.e-02
MOGWO	1	1	NaN	NaN	NaN
ZDT6					
ϵ -GMOEO	6.609.e-01	8.71.e-01	8.499.e-01	8.71.e-01	6.643.e-02
cone- ϵ -GMOEO	4.995.e-01	7.228.e-01	5.790.e-01	5.551.e-01	7.941.e-02
MOPSO	5.203.e-01	8.709.e-01	8.356.e-01	8.708.e-01	1.108.e-01
MOGWO	8.685.e-01	8.709.e-01	8.702.e-01	8.709.e-01	1.015.e-03

**Figure 3.** Pareto front obtained by the ϵ -GMOEO dominance, cone- ϵ -GMOEO dominance, MOPSO, and MOGWO methods for the test function ZDT1.

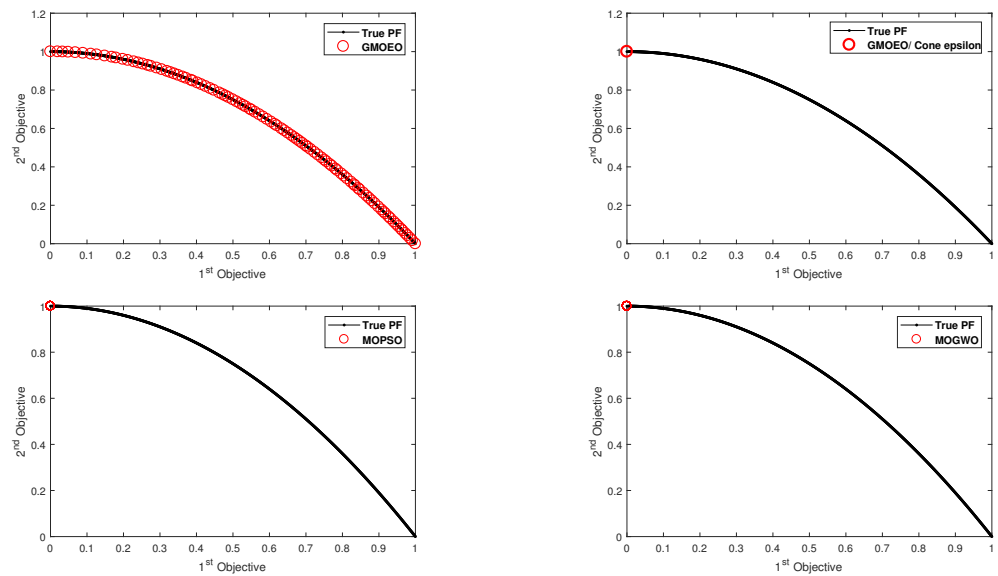


Figure 4. Pareto front obtained by the ϵ -GMOEO dominance, cone- ϵ -GMOEO dominance, MOPSO, and MOGWO methods for the test function ZDT2.

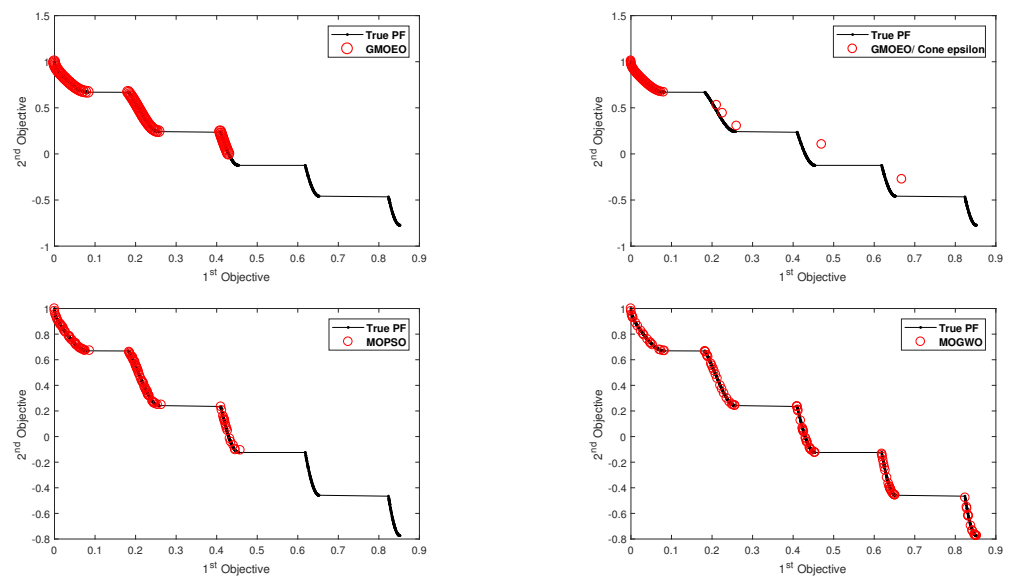


Figure 5. Pareto front obtained by the ϵ -GMOEO dominance, cone- ϵ -GMOEO dominance, MOPSO, and MOGWO methods for the test function ZDT3.

To summarize the ZDT test function results, the proposed GMOEO method achieved better outcomes when compared to the other algorithms for IGD, SP, and MS. In particular, ϵ -GMOEO was able to outperform the selected benchmarking algorithms, which proved that GMOEO was highly competitive by reaching 100% of the final non-dominated solutions within four different test functions. The metric of convergence IGD was in addition to the diversity, spread, and coverage analysis showing a 100% success rate, and this was the case within all the five different test functions.

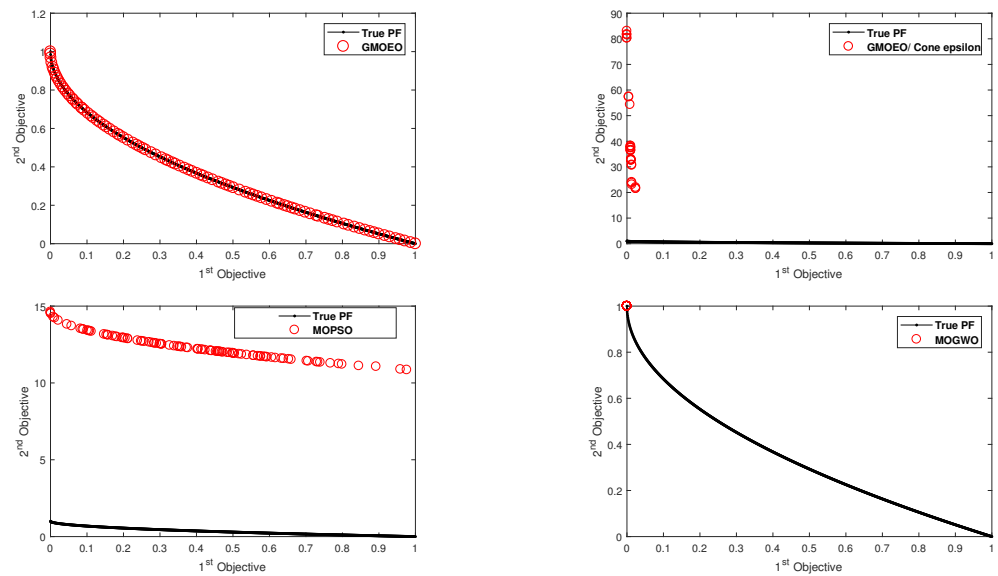


Figure 6. Pareto front obtained by the ϵ -GMOEO dominance, cone- ϵ -GMOEO dominance, MOPSO, and MOGWO methods for the test function ZDT4.

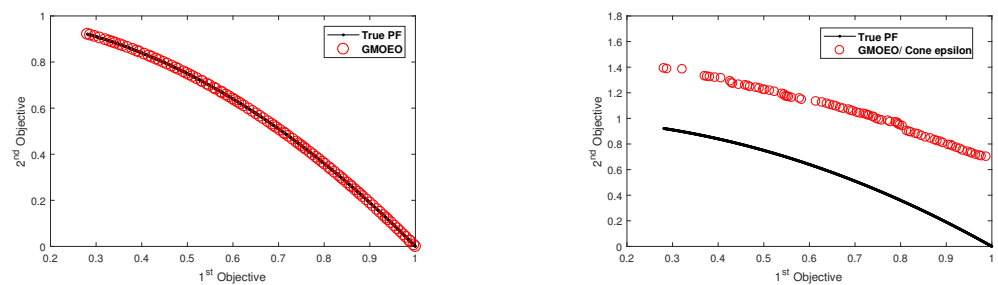


Figure 7. Cont.

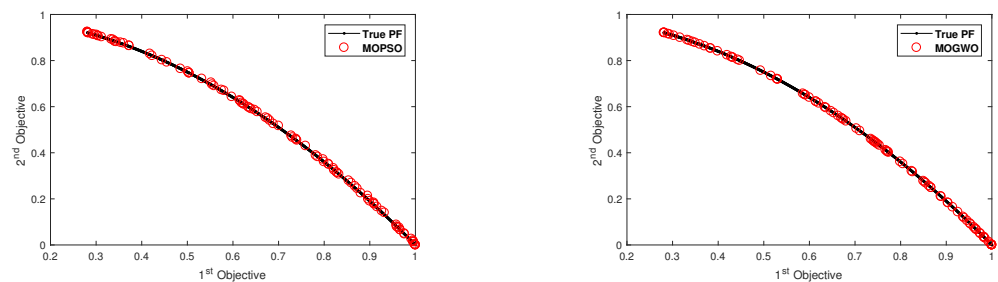


Figure 7. Pareto front obtained by the ϵ -GMOEO dominance, cone- ϵ -GMOEO dominance, MOPSO, and MOGWO methods for the test function ZDT6.

4.2. Results of the DTLZ Test Functions

The statistical results for the metrics IGD, SP, and MS for the second test function of DTLZ are summarized in Tables 5–7, and the qualitative significance of the results is illustrated in Figures 8–14. As shown in Table 5, which represents the IGD metric, the proposed ϵ -GMOEO method had the best convergence when compared to the other algorithms. It was able to outperform them in five-in-seven test functions, i.e., DTLZ2 and DTLZ4–7, and it ranked second after MOPSO for the test functions DTLZ1 and DTLZ3. This indicated ϵ -GMOEO was able to compete with well-known algorithms in terms of the convergence in challenging test functions, while cone- ϵ -GMOEO instead ranked last. Figures 8–14 plot the IGD metric results, where the ϵ -GMOEO method also showed efficient convergence. Furthermore, Tables 6 and 7 show the diversity metrics of SP and MS,

respectively. The presented results in Tables 6 and 7 indicated that ϵ -GMOEO outperformed cone- ϵ -GMOEO, MOPSO, and MOGWO in five of the test functions (e.g., DTLZ1, DTLZ3, DTLZ5–7) in terms of spacing by obtaining the best average SP value. This confirmed the efficient distribution of the non-dominated solutions over all the true Pareto solutions, as illustrated in Figures 8–14. For DTLZ2 and DTLZ4, cone- ϵ -GMOEO ranked first. For the MS metric, the reported results also showed that ϵ -GMOEO provided better diversity than the other benchmarking algorithms. Furthermore, Figures 8–14 show the front produced by ϵ -GMOEO was well extended and better distributed over the Pareto front, as shown in Figure 11, when compared to the other algorithms. The ϵ -GMOEO approach was able to converge and attain efficient distribution and diversity. Moreover, in Figure 14, when compared to the other algorithms, the front generated for DTLZ7 was well distributed and was able to converge and attain all the hyper-planes.

Therefore, the proposed ϵ -GMOEO dominance was very competitive and reliable. It was based on high performance and robustness as indicated by the statistical results for IGD, SP, and MS for the challenging DTLZ test function. In particular, the ϵ -GMOEO method was able to outperform the selected benchmarking algorithms by reaching 100% of the final non-dominated solutions within six different test functions and that for the metric of convergence IGD. In addition, the diversity, spread, and coverage analysis showed a 100% success rate and this was the case for all the five different test functions.

Table 5. Results for the IGD of the DTLZ-series test functions.

Algorithm	Best	Worst	Average	Median	Std
DTLZ1					
ϵ -GMOEO	1.273.e-01	1.41.e-01	1.295.e-01	1.282.e-01	4.338.e-03
cone- ϵ -GMOEO	2.575.e-01	5.078.e-01	3.999.e-01	4.173.e-01	8.470.e-02
MOPSO	1.711.e-02	7.506.e-02	4.281.e-02	3.886.e-02	2.176.e-02
MOGWO	1.095.e-01	2.167.e-01	1.512.e-01	1.359.e-01	3.637.e-02
DTLZ2					
ϵ -GMOEO	1.164.e-03	1.535.e-03	1.313.e-03	1.295.e-03	1.045.e-04
cone- ϵ -GMOEO	2.192.e-003	3.459.e-03	2.627.e-03	2.451.e-03	4.072.e-04
MOPSO	3.908.e-03	4.957.e-03	4.280.e-03	4.243.e-03	3.680.e-04
MOGWO	1.838.e-03	1.964.e-03	1.884.e-03	1.879.e-03	4.275.e-05
DTLZ3					
ϵ -GMOEO	1.884	2.167	2.058	2.161	1.366.e-01
cone- ϵ -GMOEO	2.611	3.7	3.129	3.102	3.188.e-01
MOPSO	3.036.e-01	1.323	7.705.e-01	7.449.e-01	3.355.e-01
MOGWO	1.680	3.184	2.403	2.427	4.183.e-01
DTLZ4					
ϵ -GMOEO	9.969.e-04	1.249.e-03	1.065.e-03	1.038.e-03	7.641.E-5
cone- ϵ -GMOEO	6.365.e-03	1.275.e-02	8.881.e-03	8.155.e-03	2.018.e-03
MOPSO	1.384.e-03	1.388.e-02	1.085.e-02	1.388.e-02	4.972.e-03
MOGWO	2.534.e-03	9.658.e-03	4.148.e-03	2.922.e-03	2.747.e-03
DTLZ5					
ϵ -GMOEO	7.543.e-05	7.973.e-05	7.757.e-05	7.694.e-05	1.651.e-06
cone- ϵ -GMOEO	2.333.e-03	3.163.e-03	2.679.e-03	2.662.e-03	2.522.e-04
MOPSO	1.318.e-04	8.166.e-04	3.342.e-04	2.498.e-04	2.222.e-04
MOGWO	4.979.e-04	7.256.e-04	6.089.e-04	6.028.e-04	7.248.e-05

Table 5. Cont.

Algorithm	Best	Worst	Average	Median	Std
DTLZ6					
ϵ -GMOEO	7.456.e-05	7.876.e-05	7.654.e-05	7.651.e-05	1.250.e-06
cone- ϵ -GMOEO	8.665.e-05	9.938.e-04	6.114.e-04	6.515.e-04	3.044.e-04
MOPSO	1.338.e-02	7.989.e-02	5.544.e-02	5.785.e-02	1.957.e-02
MOGWO	1.303.e-04	1.861.e-04	1.601.e-04	1.588.e-04	1.798.e-05
DTLZ7					
ϵ -GMOEO	1.37.e-03	1.968.e-03	1.554.e-03	1.510.e-03	1.928.e-04
cone- ϵ -GMOEO	7.949.e-03	1.812.e-02	1.498.e-02	1.624.e-02	4.033.e-03
MOPSO	1.086.e-02	1.148.e-02	1.117.e-02	1.109.e-02	2.115.e-04
MOGWO	8.084.e-04	1.812.e-02	9.516.e-03	7.817.e-03	6.555.e-03

Table 6. Results for the SP of the DTLZ-series test functions.

Algorithm	Best	Worst	Average	Median	Std
DTLZ1					
ϵ -GMOEO	2.1e-02	3.904.e-02	2.943.e-02	2.814.e-02	4.925.e-03
cone- ϵ -GMOEO	3.517.e-02	5.169.e-02	4.496.e-02	4.485.e-02	5.094.e-03
MOPSO	3.191.e-02	5.082.e-01	1.106.e-01	5.092.e-02	1.486.e-01
MOGWO	4.803.e-02	9.696.e-02	7.15.e-02	6.814.e-02	1.908.e-02
DTLZ2					
ϵ -GMOEO	3.345.e-02	4.124.e-02	3.697.e-02	3.683.e-02	2.927.e-03
cone- ϵ -GMOEO	3.190.e-02	4.45.e-02	3.934.e-02	4.067.e-02	4.206.e-03
MOPSO	2.320.e-02	3.605.e-02	2.968.e-02	2.979.e-02	3.421.e-03
MOGWO	3.263.e-02	4.967.e-02	3.879.e-02	3.698.e-02	5.177.e-03
DTLZ3					
ϵ -GMOEO	3.277.e-02	6.229.e-02	4.451.e-02	4.211.e-02	9.273.e-03
cone- ϵ -GMOEO	3.615.e-02	1.058.e-01	7.914.e-02	8.618.e-02	2.154.e-02
MOPSO	3.843.e-02	2.195.e-01	7.152.e-02	5.334.e-02	5.492.e-02
MOGWO	4.094.e-02	1.153.e-01	6.445.e-02	6.222.e-02	2.359.e-02
DTLZ4					
ϵ -GMOEO	3.018.e-02	4.376.e-02	3.828.e-02	3.911.e-02	4.558.e-03
cone- ϵ -GMOEO	2.456.e-02	2.776.e-01	1.295.e-01	1.083.e-01	8.270.e-02
MOPSO	0	5.164.e-01	1.089.e-01	9.529.e-03	2.018.e-01
MOGWO	4.672.e-02	1.879.e-01	7.417.e-02	5.154.e-02	5.087.e-02
DTLZ5					
ϵ -GMOEO	4.929.e-03	6.434.e-03	5.559.e-03	5.556.e-03	5.e-04
cone- ϵ -GMOEO	1.421.e-02	4.624.e-02	2.753.e-02	2.626.e-02	1.021.e-02
MOPSO	8.8222.e-03	1.826.e-02	1.232.e-02	1.161.e-02	3.170.e-03
MOGWO	1.431.e-02	2.18.e-02	1.726.e-02	1.627.e-02	3.060.e-03
DTLZ6					
ϵ -GMOEO	4.55.e-03	5.833.e-03	5.19.e-03	5.249.e-03	3.658.e-04
cone- ϵ -GMOEO	6.183.e-03	5.302.e-02	3.313.e-02	3.773.e-02	1.965.e-02
MOPSO	2.3831.e-02	5.012.e-02	3.387.e-02	3.204.e-02	8.467.e-03
MOGWO	8.136.e-03	1.317.e-02	1.12.e-02	1.120.e-02	1.579.e-03

Table 6. Cont.

Algorithm	Best	Worst	Average	Median	Std
DTLZ7					
ϵ -GMOEO	1.285.e-02	2.761.e-02	1.789.e-02	1.789.e-02	4.169.e-03
cone- ϵ -GMOEO	0	3.188.e-01	NaN	NaN	NaN
MOPSO	2.666.e-02	4.111.e-02	3.225.e-02	3.065.e-02	4.536.e-03
MOGWO	0	2.814.e-02	5.187.e-03	0	1.098.e-02

Table 7. Results for the MS of the DTLZ-series test functions.

Algorithm	Best	Worst	Average	Median	Std
DTLZ1					
ϵ -GMOEO	3.796.e-02	6.549.e-02	6.08.e-02	6.251.e-02	8.11.e-03
cone- ϵ -GMOEO	1.089.e-02	1.629.e-02	1.387.e-02	1.412.e-02	1.849.e-03
MOPSO	3.376.e-02	6.336.e-01	2.424.e-01	1.123.e-01	2.578.e-01
MOGWO	2.511.e-02	3.718.e-02	3.220.e-02	3.271.e-02	3.357.e-03
DTLZ2					
ϵ -GMOEO	9.997.e-01	9.999.e-01	9.998.e-01	9.998.e-01	7.512.e-05
cone- ϵ -GMOEO	5.461.e-01	7.966.e-01	6.866.e-01	6.948.e-01	8.273.e-02
MOPSO	6.487.e-01	7.579.e-01	7.175.e-01	7.349.e-01	3.992.e-02
MOGWO	7.725.e-01	9.143.e-01	8.572.e-01	8.614.e-01	4.087.e-02
DTLZ3					
ϵ -GMOEO	5.889.e-03	6.453.e-03	6.339.e-03	6.430.e-03	1.790.e-04
cone- ϵ -GMOEO	2.333.e-03	3.919.e-03	2.984.e-03	2.878.e-03	4.766.e-04
MOPSO	1.734.e-03	4.815.e-02	9.163.E-3	3.639.e-03	1.452.e-02
MOGWO	4.371.e-03	5.321.e-03	4.777.e-03	4.673.e-03	3.066.e-04
DTLZ4					
ϵ -GMOEO	9.997.e-01	9.999.e-01	9.998.e-01	9.999.e-01	7.13.E-5
cone- ϵ -GMOEO	7.082.e-01	8.945.e-01	8.079.e-01	7.974.e-01	5.326.e-02
MOPSO	7.071.e-01	9.335.e-01	NaN	NaN	NaN
MOGWO	8.753.e-01	9.806.e-01	9.123.e-01	9.042.e-01	3.550. e-02
DTLZ5					
ϵ -GMOEO	1	1	1	1	0
cone- ϵ -GMOEO	9.216.e-01	1	9.88.e-01	1	2.568.e-02
MOPSO	7.570.e-01	1	9.66.e-01	1	7.636.e-02
MOGWO	1	1	1	1	0
DTLZ6					
ϵ -GMOEO	1	1	1	1	0
cone- ϵ -GMOEO	1	1	1	1	0
MOPSO	6.528.e-02	7.962.e-01	1.974.e-01	1.282.e-01	2.164.e-01
MOGWO	1 1 1 1 0				
DTLZ7					
ϵ -GMOEO	1	1	1	1	0
cone- ϵ -GMOEO	9.998.e-01	1	NaN	NaN	NaN
MOPSO	1	1	1	1	0
MOGWO	1	1	NaN	NaN	NaN

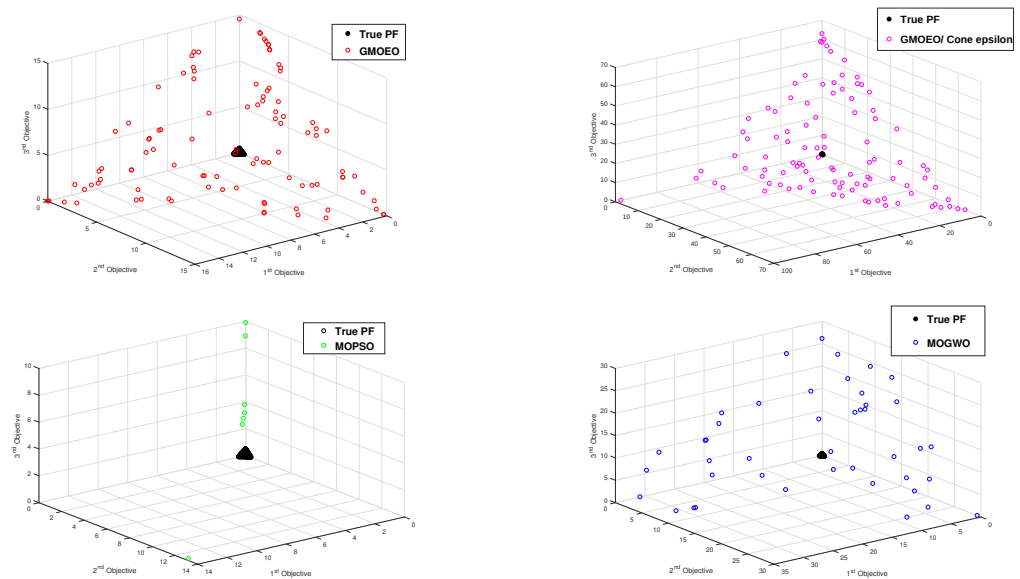


Figure 8. Pareto front obtained by the ϵ -GMOEO dominance, cone- ϵ -GMOEO dominance, MOPSO, and MOGWO methods for the test function DTLZ1.

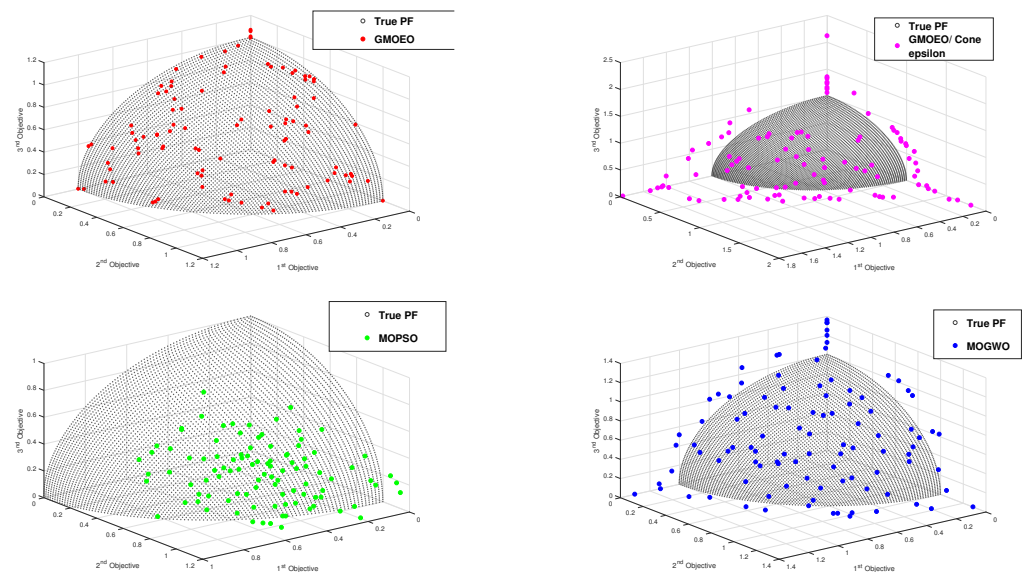


Figure 9. Pareto front obtained by the ϵ -GMOEO dominance, cone- ϵ -GMOEO dominance, MOPSO, and MOGWO methods for the test function DTLZ2.

As shown in the presented figures, the convergence, coverage, and excellent distribution of the solutions along the Pareto optimal indicated the efficient performance of the proposed ϵ -GMOEO method.

To further support the findings and to confirm the performance of the proposed GMOEO algorithm, the Wilcoxon rank-sum test [54] was considered. The analysis was conducted according to the p_{value} and the level of significance α among the compared algorithms. The Wilcoxon rank-sum test would assist in establishing the significant differences according to the null-hypothesis (H_0): If $p_{value} > \alpha$, then no significant difference between the results of the algorithms could be found. However, if $p_{value} \leq \alpha$, then a significant difference was found (H_1).

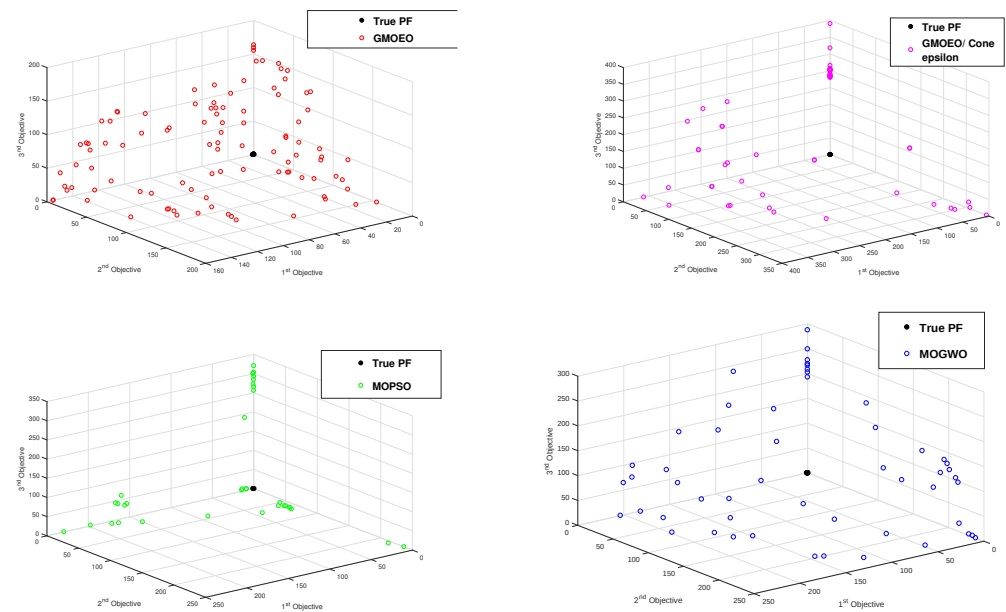


Figure 10. Pareto front obtained by the ϵ -GMOEO dominance, cone- ϵ -GMOEO dominance, MOPSO, and MOGWO methods for the test function DTLZ3.

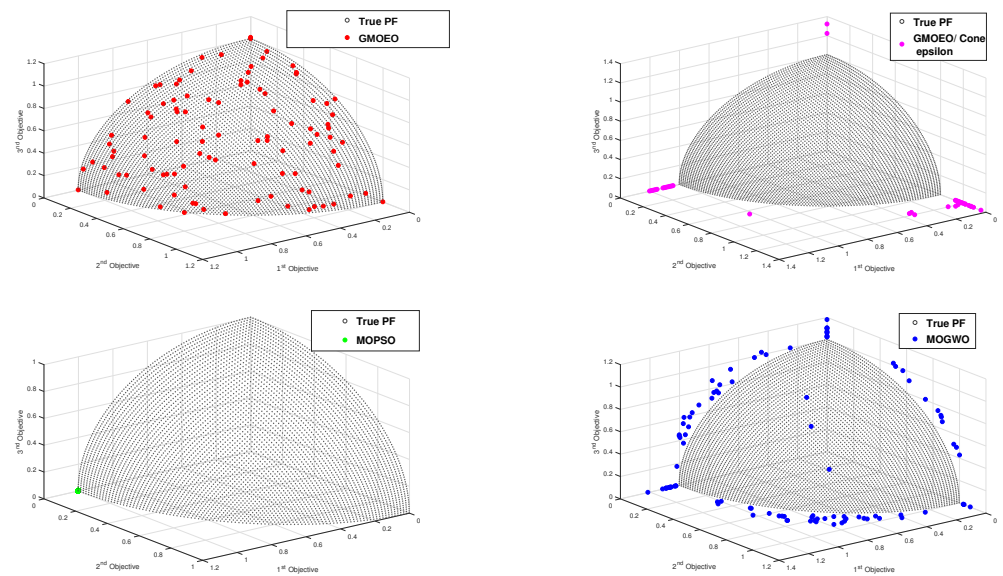


Figure 11. Pareto front obtained by the ϵ -GMOEO dominance, cone- ϵ -GMOEO dominance, MOPSO, and MOGWO methods for the test function DTLZ4.

Therefore, we evaluated the p_{value} of the IGD and SP metrics that were obtained after ten operations by comparing each pair of ϵ -GMOEO and cone- ϵ -GMOEO, and MOPSO and MOGWO for each test function with a significance of $= 5\%$. According to the p_{value} results for the IGD metric shown in Table 8, the ϵ -GMOEO outperformed all the benchmarking algorithms in all the test functions, with both bi- and tri-objectives. As shown in Table 8, the $p_{value} \leq \alpha$ indicated that the hypothesis H_1 was confirmed. Furthermore, the results showed that the proposed GMOEO was able to converge toward the Pareto optimal more efficiently than the other algorithms, including cone- ϵ -GMOEO. According to the p_{value} results of the SP metric shown in Table 9, the ϵ -GMOEO method outperformed all other algorithms in 98% of the test functions.

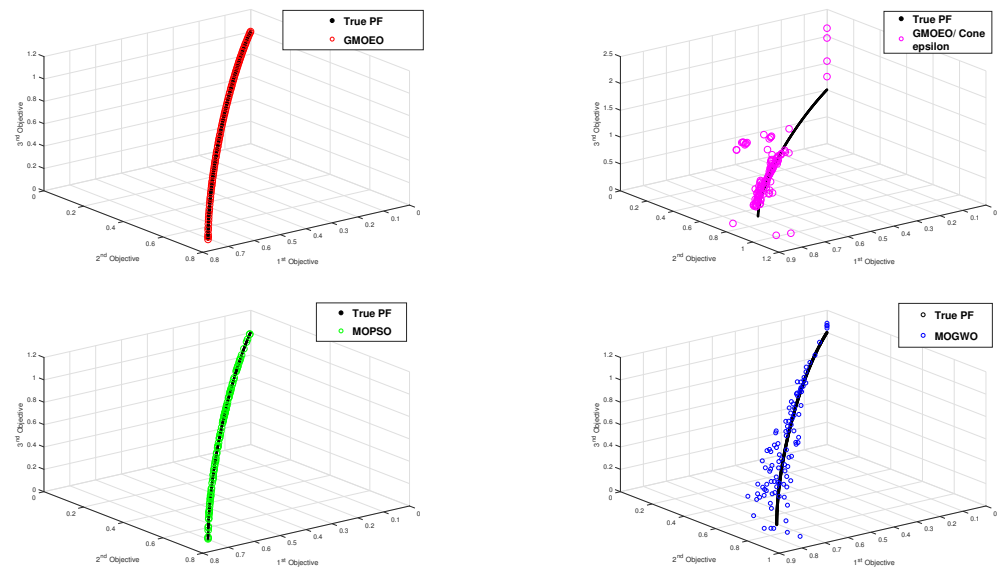


Figure 12. Pareto front obtained by the ϵ -GMOEO dominance, cone- ϵ -GMOEO dominance, MOPSO, and MOGWO methods for the test function DTLZ5.

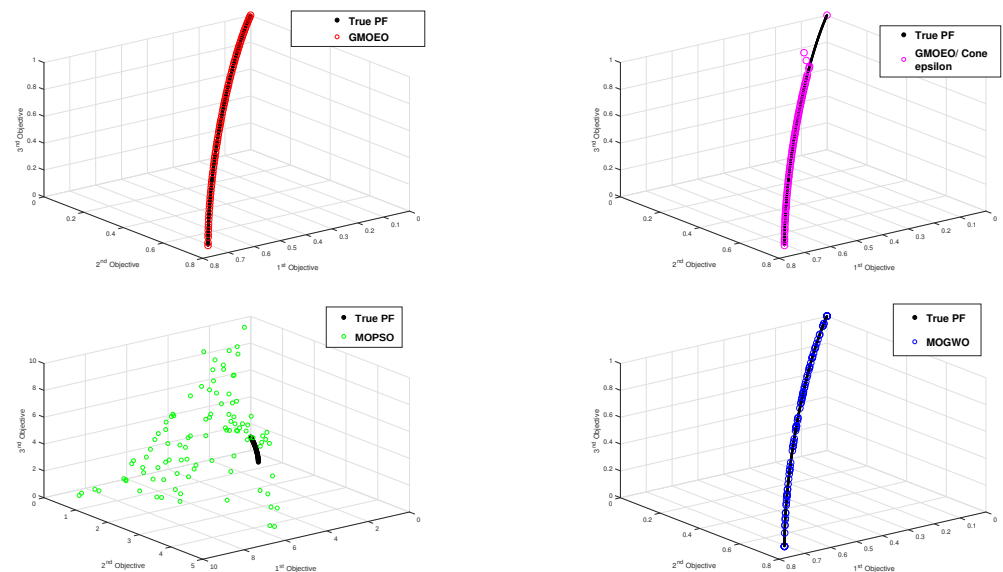


Figure 13. Pareto front obtained by the ϵ -GMOEO dominance, cone- ϵ -GMOEO dominance, MOPSO, and MOGWO methods for the test function DTLZ6.

Therefore, ϵ -GMOEO has better diversity and spread. In summary, the results of the Wilcoxon rank-sum test proved that ϵ -GMOEO was highly competitive when compared to existing algorithms as it did the best 100% of the time. In addition, the performance of ϵ -GMOEO was capable of nearly reaching the Pareto front reliably. Finally, we confirmed that the archive population and update strategies were effective for increasing the reliability of GMOEO, along with the application of the crowding distance and FNS methods. Furthermore, ϵ -dominance was a better option for updating the archive when compared to cone- ϵ -dominance, even though the latter had outperformed some of the benchmarking algorithms. However, it was not as effective as ϵ -dominance.

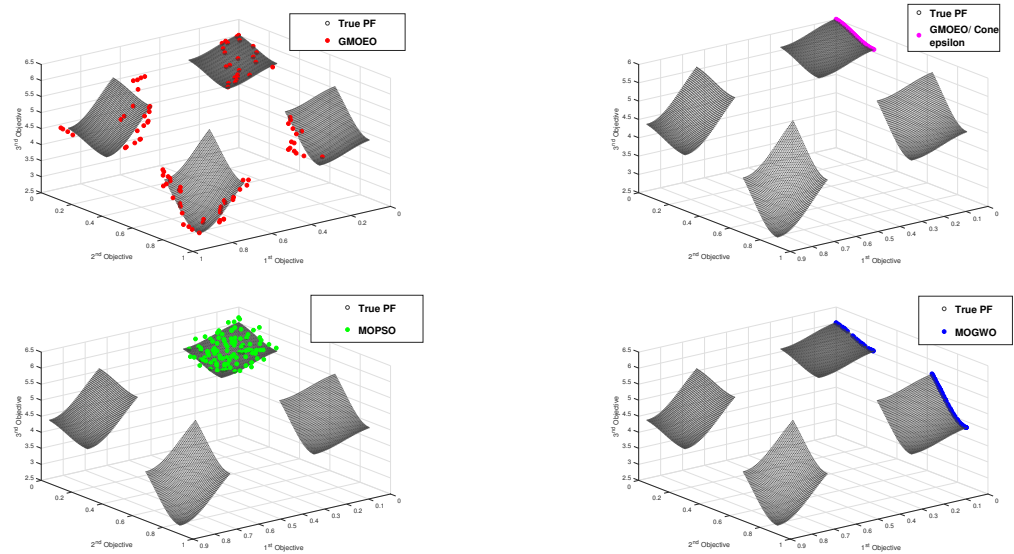


Figure 14. Pareto front obtained by the ϵ -GMOEO dominance, cone- ϵ -GMOEO dominance, MOPSO, and MOGWO methods for the test function DTLZ7.

Table 8. The p_{value} results for the Wilcoxon rank-sum test of the IGD metric.

	cone- ϵ -GMOEO	MOPSO	MOGWO
ZDT1	1.8267.e-4	1.8267.e-4	1.8267.e-4
ZDT2	1.7167.e-4	8.74498.e-5	1.1067.e-4
ZDT3	2.8272.e-3	1.8267.e-4	1.8267.e-4
ZDT4	1.8267.e-4	1.8267.e-4	2.4375.e-4
ZDT6	1.8267.e-4	1.8267.e-4	1.8267.e-4
DTLZ1	1.8267.e-4	1.8267.e-4	0.2413
DTLZ2	1.8267.e-4	1.8267.e-4	1.8267.e-4
DTLZ3	1.8267.e-4	1.8267.e-4	1.1329.e-2
DTLZ4	1.8267.e-4	1.8267.e-4	1.8267.e-4
DTLZ5	1.8267.e-4	1.8267.e-4	1.8267.e-4
DTLZ6	1.8267.e-4	1.8267.e-4	1.8267.e-4
DTLZ7	1.8165.e-4	1.8267.e-4	2.5525.e-2

Table 9. The p_{value} for the Wilcoxon rank-sum test of the SP metric.

	cone- ϵ -GMOEO	MOPSO	MOGWO
ZDT1	1.8267.e-4	1.8267.e-4	1.8267.e-4
ZDT2	3.0303.e-2	1.7451.e-3	2.1226.e-2
ZDT3	1.8267.e-4	2.4612.e-4	1.8267.e-4
ZDT4	1.8267.e-4	5.8283.e-4	4.3420.e-3
ZDT6	2.2022.e-3	2.8272.e-3	2.8272.e-3
DTLZ1	2.4612.e-4	7.6853.e-4	1.8267.e-4
DTLZ2	0.1619	7.6853.e-4	0.5205
DTLZ3	2.8272.e-3	0.1404	3.7635.e-2
DTLZ4	4.5863.e-3	0.4693	1.8267.e-4
DTLZ5	1.8267.e-4	1.8267.e-4	1.8267.e-4
DTLZ6	1.8267.e-4	1.8267.e-4	1.8267.e-4
DTLZ7	0.2997	3.2983.e-4	1.7217.e-2

5. Conclusions

In this study, we proposed a guided multi-objective equilibrium optimizer (GMOEO) to solve MOPs. The GMOEO algorithm was developed by extending the equilibrium

optimizer via an external archive to guide the population toward the Pareto optimal. Furthermore, fast non-dominated sorting and crowding distance methods were introduced into the GMOEO algorithm in order to control the quality of the solutions and to ensure the diversity and convergence factors. More importantly, we explored two relations, i.e., ϵ -dominance and cone- ϵ -dominance, for the purposes of managing the archive population's best non-dominated solutions. In addition, a candidate population was proposed to aid in updating the archive. A benchmarking study was conducted with the well-known MOPSO and MOGWO algorithms on 12 widely used benchmark functions. The obtained results indicated that the proposed GMOEO is a powerful and competitive tool with high efficacy. It was able to outperform other algorithms. Moreover, according to the statistical and graphical results for this algorithm, we were able to realize ϵ -dominance as a better strategy for updating the archive when compared to the cone- ϵ -dominance. Finally, the proposed GMOEO was a reliable tool for solving multi-objective optimization problems. GMOEO could also be a efficient tool for solving several engineering optimization problems, such as the design of pressure vessels, vibrating platform, and gear trains, as well as other problems—the results showed GMOEO's potential for such applications.

Author Contributions: Conceptualization, A.B. (Abderrahouf Bouziane) and A.B. (Adel Binbusayyis); methodology, N.E.C. and A.A. (Abdelouahab Attia); software, N.E.C.; validation, M.H.; formal analysis, M.H., A.A. (Abdelouahab Attia), and A.A. (Abed Alanazi); writing—original draft preparation, N.E.C. and A.A. (Abdelouahab Attia); writing—review and editing, M.H., A.A. (Abed Alanazi), and A.B. (Adel Binbusayyis); project administration, A.B. (Adel Binbusayyis) and A.A. (Abdelouahab Attia); funding acquisition, M.H., A.A. (Abed Alanazi) and A.B. (Adel Binbusayyis). All authors have read and agreed to the published version of the manuscript.

Funding: Funding was received from Prince Sattam bin Abdulaziz University (project number (PSAU/2023/R/1444)).

Data Availability Statement: Not applicable.

Acknowledgments: This study was supported via funding from Prince Sattam bin Abdulaziz University (project number (PSAU/2023/R/1444)).

Conflicts of Interest: The authors declare no conflicts of interests.

References

1. Dahou, A.; Chelloug, S.A.; Alduailij, M.; Elaziz, M.A. Improved Feature Selection Based on Chaos Game Optimization for Social Internet of Things with a Novel Deep Learning Model. *Mathematics* **2023**, *11*, 1032. [\[CrossRef\]](#)
2. Mohamed, A.A.; Abdellatif, A.D.; Alburaikan, A.; Khalifa, H.A.E.W.; Elaziz, M.A.; Abualigah, L.; AbdelMouty, A.M. A novel hybrid arithmetic optimization algorithm and salp swarm algorithm for data placement in cloud computing. *Soft Comput.* **2023**, *27*, 5769–5780. [\[CrossRef\]](#)
3. Vijaya Bhaskar, K.; Ramesh, S.; Karunanithi, K.; Raja, S. Multi Objective Optimal Power Flow Solutions using Improved Multi Objective Mayfly Algorithm (IMOMA). *J. Circuits Syst. Comput.* **2023**. [\[CrossRef\]](#)
4. Perera, J.; Liu, S.H.; Mernik, M.; Črepinšek, M.; Ravber, M. A Graph Pointer Network-Based Multi-Objective Deep Reinforcement Learning Algorithm for Solving the Traveling Salesman Problem. *Mathematics* **2023**, *11*, 437. [\[CrossRef\]](#)
5. Zhang, Z.; Gao, S.; Lei, Z.; Xiong, R.; Cheng, J. Pareto Dominance Archive and Coordinated Selection Strategy-Based Many-Objective Optimizer for Protein Structure Prediction. *IEEE/ACM Trans. Comput. Biol. Bioinform.* **2023**, *20*, 2328–2340. [\[CrossRef\]](#)
6. De, S.; Dey, S.; Bhattacharyya, S. *Recent Advances in Hybrid Metaheuristics for Data Clustering*; John Wiley & Sons: Hoboken, NJ, USA, 2020.
7. Bhattacharyya, S. *Hybrid Computational Intelligent Systems: Modeling, Simulation and Optimization*; CRC Press: Boca Raton, FL, USA, 2023.
8. Bhattacharyya, S.; Banerjee, J.S.; De, D. *Confluence of Artificial Intelligence and Robotic Process Automation*; Springer: Berlin/Heidelberg, Germany, 2023.
9. Mahmoodabadi, M. An optimal robust fuzzy adaptive integral sliding mode controller based upon a multi-objective grey wolf optimization algorithm for a nonlinear uncertain chaotic system. *Chaos Solitons Fractals* **2023**, *167*, 113092. [\[CrossRef\]](#)
10. Chalabi, N.E.; Attia, A.; Bouziane, A.; Hassaballah, M. An improved marine predator algorithm based on Epsilon dominance and Pareto archive for multi-objective optimization. *Eng. Appl. Artif. Intell.* **2023**, *119*, 105718. [\[CrossRef\]](#)
11. Feng, X.; Pan, A.; Ren, Z.; Fan, Z. Hybrid driven strategy for constrained evolutionary multi-objective optimization. *Inf. Sci.* **2022**, *585*, 344–365. [\[CrossRef\]](#)

12. Houssein, E.H.; Saad, M.R.; Hashim, F.A.; Shaban, H.; Hassaballah, M. Lévy flight distribution: A new metaheuristic algorithm for solving engineering optimization problems. *Eng. Appl. Artif. Intell.* **2020**, *94*, 103731. [\[CrossRef\]](#)
13. Deb, K.; Pratap, A.; Agarwal, S.; Meyarivan, T. A fast and elitist multiobjective genetic algorithm: NSGA-II. *IEEE Trans. Evol. Comput.* **2002**, *6*, 182–197. [\[CrossRef\]](#)
14. Deb, K.; Jain, H. An evolutionary many-objective optimization algorithm using reference-point-based nondominated sorting approach, part I: Solving problems with box constraints. *IEEE Trans. Evol. Comput.* **2013**, *18*, 577–601. [\[CrossRef\]](#)
15. Knowles, J.D.; Corne, D.W. M-PAES: A memetic algorithm for multiobjective optimization. In Proceedings of the Congress on Evolutionary Computation CEC00 (Cat. No. 00TH8512), La Jolla, CA, USA, 16–19 July 2000; Volume 1, pp. 325–332.
16. Zitzler, E.; Laumanns, M.; Thiele, L. SPEA2: Improving the strength Pareto evolutionary algorithm. *TIK-Report* **2001**, *103*. [\[CrossRef\]](#)
17. Daliri, A.; Asghari, A.; Azgomi, H.; Alimoradi, M. The water optimization algorithm: A novel metaheuristic for solving optimization problems. *Appl. Intell.* **2022**, *52*, 17990–18029. [\[CrossRef\]](#)
18. Qin, S.; Pi, D.; Shao, Z.; Xu, Y.; Chen, Y. Reliability-Aware Multi-Objective Memetic Algorithm for Workflow Scheduling Problem in Multi-Cloud System. *IEEE Trans. Parallel Distrib. Syst.* **2023**, *34*, 1343–1361. [\[CrossRef\]](#)
19. Abd Elaziz, M.; Abualigah, L.; Issa, M.; Abd El-Latif, A.A. Optimal parameters extracting of fuel cell based on Gorilla Troops Optimizer. *Fuel* **2023**, *332*, 126162. [\[CrossRef\]](#)
20. Dutta, T.; Bhattacharyya, S.; Panigrahi, B.K. Multilevel Quantum Evolutionary Butterfly Optimization Algorithm for Automatic Clustering of Hyperspectral Images. In Proceedings of the 3rd International Conference on Artificial Intelligence and Computer Vision, Taiyuan, China, 26–28 May 2023; pp. 524–534.
21. Zhang, Q.; Li, H. MOEA/D: A multiobjective evolutionary algorithm based on decomposition. *IEEE Trans. Evol. Comput.* **2007**, *11*, 712–731. [\[CrossRef\]](#)
22. Deb, K.; Mohan, M.; Mishra, S. Evaluating the ϵ -domination based multi-objective evolutionary algorithm for a quick computation of Pareto-optimal solutions. *Evol. Comput.* **2005**, *13*, 501–525. [\[CrossRef\]](#)
23. Ma, X.; Qi, Y.; Li, L.; Liu, F.; Jiao, L.; Wu, J. MOEA/D with uniform decomposition measurement for many-objective problems. *Soft Comput.* **2014**, *18*, 2541–2564. [\[CrossRef\]](#)
24. Tan, Y.Y.; Jiao, Y.C.; Li, H.; Wang, X.K. MOEA/D-SQA: A multi-objective memetic algorithm based on decomposition. *Eng. Optim.* **2012**, *44*, 1095–1115. [\[CrossRef\]](#)
25. Qiao, K.; Liang, J.; Yu, K.; Wang, M.; Qu, B.; Yue, C.; Guo, Y. A Self-Adaptive Evolutionary Multi-Task Based Constrained Multi-Objective Evolutionary Algorithm. *IEEE Trans. Emerg. Top. Comput. Intell.* **2023**. [\[CrossRef\]](#)
26. Wang, Q.; Gu, Q.; Chen, L.; Guo, Y.; Xiong, N. A MOEA/D with global and local cooperative optimization for complicated bi-objective optimization problems. *Appl. Comput.* **2023**, *137*, 110162. [\[CrossRef\]](#)
27. Eberhart, R.; Kennedy, J. A new optimizer using particle swarm theory. In Proceedings of the Sixth International Symposium on Micro Machine and Human Science, Nagoya, Japan, 4–6 October 1995; pp. 39–43.
28. Rabbani, M.; Oladad-Abbasabady, N.; Akbarian-Saravi, N. Ambulance routing in disaster response considering variable patient condition: NSGA-II and MOPSO algorithms. *J. Ind. Manag. Optim.* **2022**, *18*, 1035–1062. [\[CrossRef\]](#)
29. Ray, T.; Liew, K. A swarm metaphor for multiobjective design optimization. *Eng. Optim.* **2002**, *34*, 141–153. [\[CrossRef\]](#)
30. Coello, C.A.C.; Pulido, G.T.; Lechuga, M.S. Handling multiple objectives with particle swarm optimization. *IEEE Trans. Evol. Comput.* **2004**, *8*, 256–279. [\[CrossRef\]](#)
31. Dorigo, M.; Di Caro, G. Ant colony optimization: A new meta-heuristic. In Proceedings of the Congress on Evolutionary Computation-CEC99 (Cat. No. 99TH8406), Washington, DC, USA, 6–9 July 1999; Volume 2, pp. 1470–1477.
32. Kaveh, M.; Mesgari, M.S.; Saeidian, B. Orchard Algorithm (OA): A new meta-heuristic algorithm for solving discrete and continuous optimization problems. *Math. Comput. Simul.* **2023**, *208*, 95–135. [\[CrossRef\]](#)
33. Rada-Vilela, J.; Chica, M.; Cordon, Ó.; Damas, S. A comparative study of multi-objective ant colony optimization algorithms for the time and space assembly line balancing problem. *Appl. Soft Comput.* **2013**, *13*, 4370–4382. [\[CrossRef\]](#)
34. Pu, X.; Song, X.; Tan, L.; Zhang, Y. Improved ant colony algorithm in path planning of a single robot and multi-robots with multi-objective. *Evol. Intell.* **2023**. [\[CrossRef\]](#)
35. Zhang, D.; Luo, R.; Yin, Y.b.; Zou, S.I. Multi-objective path planning for mobile robot in nuclear accident environment based on improved ant colony optimization with modified A. *Nucl. Eng. Technol.* **2023**, *55*, 1838–1854. [\[CrossRef\]](#)
36. Flor-Sánchez, C.O.; Reséndiz-Flores, E.O.; García-Calvillo, I.D. Kernel-based hybrid multi-objective optimization algorithm (KHMO). *Inf. Sci.* **2023**, *624*, 416–434. [\[CrossRef\]](#)
37. Singh, P.; Muchahari, M.K. Solving multi-objective optimization problem of convolutional neural network using fast forward quantum optimization algorithm: Application in digital image classification. *Adv. Eng. Softw.* **2023**, *176*, 103370. [\[CrossRef\]](#)
38. Chu, S.C.; Tsai, P.W. Computational intelligence based on the behavior of cats. *Int. J. Innov. Comput. Inf. Control* **2007**, *3*, 163–173.
39. Pradhan, P.M.; Panda, G. Solving multiobjective problems using cat swarm optimization. *Expert Syst. Appl.* **2012**, *39*, 2956–2964. [\[CrossRef\]](#)
40. Mirjalili, S.; Mirjalili, S.M.; Lewis, A. Grey wolf optimizer. *Adv. Eng. Softw.* **2014**, *69*, 46–61. [\[CrossRef\]](#)
41. Mirjalili, S.; Saremi, S.; Mirjalili, S.M.; Coelho, L.d.S. Multi-objective grey wolf optimizer: A novel algorithm for multi-criterion optimization. *Expert Syst. Appl.* **2016**, *47*, 106–119. [\[CrossRef\]](#)

42. Zouache, D.; Abdelaziz, F.B.; Lefkir, M.; Chalabi, N.E.H. Guided Moth–Flame optimiser for multi-objective optimization problems. *Ann. Oper. Res.* **2021**, *296*, 877–899. [\[CrossRef\]](#)
43. Mirjalili, S. Moth-flame optimization algorithm: A novel nature-inspired heuristic paradigm. *Knowl.-Based Syst.* **2015**, *89*, 228–249. [\[CrossRef\]](#)
44. Houssein, E.H.; Çelik, E.; Mahdy, M.A.; Ghoniem, R.M. Self-adaptive Equilibrium Optimizer for solving global, combinatorial, engineering, and Multi-Objective problems. *Expert Syst. Appl.* **2022**, *195*, 116552. [\[CrossRef\]](#)
45. Faramarzi, A.; Heidarinejad, M.; Stephens, B.; Mirjalili, S. Equilibrium optimizer: A novel optimization algorithm. *Knowl.-Based Syst.* **2020**, *191*, 105190. [\[CrossRef\]](#)
46. Laumanns, M.; Thiele, L.; Deb, K.; Zitzler, E. Combining convergence and diversity in evolutionary multiobjective optimization. *Evol. Comput.* **2002**, *10*, 263–282. [\[CrossRef\]](#)
47. Batista, L.S.; Campelo, F.; Guimaraes, F.G.; Ramírez, J.A. Pareto cone ϵ -dominance: Improving convergence and diversity in multiobjective evolutionary algorithms. In Proceedings of the International Conference on Evolutionary Multi-Criterion Optimization, Ouro Preto, Brazil, 5–8 April 2011; pp. 76–90.
48. Ikeda, K.; Kita, H.; Kobayashi, S. Failure of Pareto-based MOEAs: Does non-dominated really mean near to optimal? In Proceedings of the Congress on Evolutionary Computation (Cat. No. 01TH8546), Seoul, Republic of Korea, 27–30 May 2001; Volume 2, pp. 957–962.
49. Zitzler, E.; Deb, K.; Thiele, L. Comparison of multiobjective evolutionary algorithms: Empirical results. *Evol. Comput.* **2000**, *8*, 173–195. [\[CrossRef\]](#)
50. Deb, K.; Thiele, L.; Laumanns, M.; Zitzler, E. Scalable multi-objective optimization test problems. In Proceedings of the Congress on Evolutionary Computation (Cat. No. 02TH8600), Honolulu, HI, USA, 12–17 May 2002; Volume 1, pp. 825–830.
51. Schott, J.R. Fault Tolerant Design Using Single and Multicriteria Genetic Algorithm Optimization. Ph.D Thesis, Massachusetts Institute of Technology, Cambridge, MA, USA, 1995.
52. Zitzler, E.; Thiele, L. Multiobjective optimization using evolutionary algorithms—A comparative case study. In Proceedings of the International Conference on Parallel Problem Solving From Nature, Amsterdam, The Netherlands, 27–30 September 1998; pp. 292–301.
53. Van Veldhuizen, D.A.; Lamont, G.B. Multiobjective evolutionary algorithms: Analyzing the state-of-the-art. *Evol. Comput.* **2000**, *8*, 125–147. [\[CrossRef\]](#) [\[PubMed\]](#)
54. Derrac, J.; García, S.; Molina, D.; Herrera, F. A practical tutorial on the use of nonparametric statistical tests as a methodology for comparing evolutionary and swarm intelligence algorithms. *Swarm Evol. Comput.* **2011**, *1*, 3–18. [\[CrossRef\]](#)

Disclaimer/Publisher’s Note: The statements, opinions and data contained in all publications are solely those of the individual author(s) and contributor(s) and not of MDPI and/or the editor(s). MDPI and/or the editor(s) disclaim responsibility for any injury to people or property resulting from any ideas, methods, instructions or products referred to in the content.