**Journal of Algorithms and Computation**

journal homepage: http://jac.ut.ac.ir

# Using a New Strategy in Imperialist Competitive Algorithm to Solve Multi-objective Problems (WSICA)

M. Moosapour*[1], A. Bagheri†[2] and M. J. Mahmoodabadi‡[3]

[1,2]Department of Mechanical Engineering, Guilan University, Rasht, Iran
[3]Department of Mechanical Engineering, Sirjan University of Technology, Sirjan, Iran

**ABSTRACT**

The imperialist competitive algorithm (ICA) is developed based on the socio-political process of imperialist competitions. It is an efficient approach for single-objective optimization problems. However, this algorithm fails to optimize multi-objective problems (MPOs) with conflicting objectives. This paper presents a modification of the ICA to different multi-objective problems. To improve the algorithm performance and adapt to the characteristics of MOPs, the Sigma method was used to establish the initial empires, the weighted sum approach (WSum) was employed for empire competition, and an adaptive elimination approach was used for external archiving strategy. The results indicated that the suggested algorithm had a higher performance compared

*Keyword:* Imperialist competitive algorithm , multi-objective optimization , Sigma method , meta-heuristics , weighted sum approach.

**ARTICLE INFO**

*Article history:*
Research paper
Received 16, April 2023
Received in revised form 16, May 2023
Accepted 29 May 2023
Available online 01, June 2023

AMS subject Classification: 65K10 , 68W50 , 90C59.

*Corresponding author: M. Moosapour. Email: mina.moosapour@gmail.com
†bagheri@guilan.ac.ir
‡mahmoodabadi@sirjantech.ac.ir

**Abstract continued**
to other algorithms based on diversity and convergence characteristics.

# 1 Introduction

Optimization problems have become more important because they arise naturally in most disciplines and can pose significant challenges for researchers. This class of problems can be characterized by either single- or multi-objective problems, while most of the real world optimization problems are multi-objective ones. However, multi-objective optimization problems are more difficult than single-objective problems, because there are often conflicts between various objective functions that must be optimized, simultaneously. The balance between the values of different objective functions will result in a set of optimal solutions called the Pareto front. Considering any solutions on Pareto front, no feasible solution exists in the search space that improves one or more objectives without simultaneously degrading at least one of the others. Hence, any multi-objective algorithm should aim at finding the Pareto front of these non-dominated solutions.

Optimization problems are being successfully addressed by bio-inspired algorithms such as the genetic algorithm (GA), and the particle swarm algorithm (PSO). The Imperialist competitive algorithm (ICA) is another kind of evolutionary algorithms for optimization problems that is more intelligent than biological behavior. The strategy is built upon an original idea, inspired by social and political events to develop a strong optimization strategy [4]. At present, this algorithm has been widely used to various fields, including artificial intelligence [12, 1], power electronic engineering [20], supply chain management [2, 19, 17, 21], vehicle scheduling [14, 15, 29], production process scheduling [38, 42, 24], design of thermal systems [26], design of linear induction motors [40], and design of skeletal structures [22], etc.

In recent years, a number of studies have been carried out regarding solving multi-objective optimization problems using ICA. Enaytifar [13] proposed the multi-objective imperialist competitive algorithm (MOICA). The numerical results of their proposed algorithm indicate that MOICA shows significantly higher efficiency in terms of accuracy and maintaining a diverse population of solutions compared to other prominent existing algorithms such as the NSGA-II and MOPSO. In addition, considering computational time, it is slightly faster than MOPSO and significantly performs better than NSGA-II. Ghasemi [16] presented a Gaussian Bare-bones multi-objective Imperialist Competitive Algorithm (GBICA) and its Modified version (MGBICA) for optimal electric power planning in the electric power system. In that paper, a new attraction policy was introduced in empire assimilation, in which colonies of other imperialists, apart from the strongest imperialist, randomly move toward three different directions: their own imperialist, the strongest imperialist, and both of them. Using this attraction policy, the population diversity continuously changes among imperialists, and their search power for the global optimum is greatly increased. Moreover, Sharifi [35] introduced a multi-objective modified ICA for a brushless DC motor optimization problem. In the proposed algorithm, a new step requiring all countries to move toward the best imperialist was added to the standard version of the ICA, and some reforms in this movement were also made. Their simulation results, considering efficiency and total mass as two objective functions, show the superiority of the proposed algorithm in the obtained Pareto fronts compared to the samples found by the SPEA2, NSGAIII, MOPSO, and MOICA. In addition, Nejlaoui [32] developed a hybrid multi-objective imperialist competitive algorithm and Monte Carlo method for designing a rail vehicle's robust safety under uncertain design parameters. In this robust optimization of rail vehicle safety, the derailment angle and standard deviation were considered, simultaneously. Their obtained results demonstrated that robust design appreciably reduces the sensitivity of rail vehicle safety to design parameters uncertainties comparison to design parameters deterministic. Moreover, Piroozfard [33] developed a new multi-objective ICA to solve a multi-objective job shop scheduling optimization problem with low carbon emissions, with the goal of simultaneously minimizing carbon footprint and total late work. Their numerically obtained results and comparison metrics shown the effectiveness and efficiency of the proposed multi-objective imperialist competitive algorithm in finding high-quality non-dominated schedules as compared with the MOPSO and NSGA-II. Furthermore, Khanali [Khanali[23] studied the energy flow and environmental emissions from walnut orchards in Iran's Alborz province, as well as their simultaneous optimization using a newly modified MOICA. The results obtained by them reveal that the orchardist can ensure optimal conditions with timely maintenance. In addition, this new colonial competition algorithm can not only provide the optimal pattern for walnut production but can also be used for different crops around the world. Li [34] developed an imperialist competitive algorithm with feedback in order to solve energy-efficient flexible job shop scheduling problems with transportation and sequence-dependent setup times, which is a complex multi-objective problem. In this algorithm, an assimilation and adaptive revolution mechanism with feedback are used. Meanwhile, an imperialist competition is presented for transferring solutions between empires to improve search capabilities. Luo [25] proposed a modification of the ICA to solve multi-objective optimization problems with hybrid methods. This research endeavors to adapt to the characteristics of multi-objective optimization problems by improving the mechanism for the formation of initial empires, colony allocation, and empire competition, and also to introduce an external archiving strategy.

This paper aims to develop a version of the single-objective ICA to solve problems with multiple objectives. There are three main strategies to accomplish this goal: using the Sigma method in empire formation to improve convergence, employing the WSum method in competition between empires to maintain diversity, and using an elimination approach to limit the number of non-dominated solutions in the archive. The performance of the proposed algorithm is compared using four well-known quality indicators against the six algorithms and a set of well-known high-dimensional benchmark functions. The results show that the proposed algorithm produces better solutions in terms of convergence and distribution along the Pareto front.

The remainder of this paper has the following structure. The general stages of the original ICA are briefly explained in the first section. In the second section, the mathematical model of multi-objective optimization problems is presented. The third section provides details about the proposed algorithm. The fourth section introduces the qualitative and quantitative results, performance metrics, comparison algorithms, simulating settings, and relevant discussion. Finally, the paper ends with a conclusion.

# 2 Brief Description of the ICA

Various evolutionary algorithms have been proposed by researchers to solve optimization problems. Each of these algorithms is based on a different evolutionary mechanism inspired by nature, such as genetic evolution in GA, swarm intelligence in PSO and ant colony optimization, or the material annealing process in simulated annealing. Accordingly, the ICA method is a general searching algorithm based on the socio-political relationship of countries, which was proposed by Lucas et al. [4]. This algorithm has demonstrated excellent capabilities in both convergence rate and global optimal achievement. ICA starts by generating a random population of individuals (called a country) in a similar way to other evolutionary algorithms. In the initialization step, the most powerful countries (with lower costs) are selected as imperialists, and other countries are considered to be the colonies of the imperialists. Then the colonies are divided among imperialists according to imperialist power, and empires are created. Following the division of all colonies among the imperialist countries, all colonies move toward their relevant imperialist through a process known as assimilation. If the colony becomes stronger than the relevant imperialist in the assimilation policy of the empire, the position of the imperialist and the colony will change. This algorithm's foundation is imperialist competition among these empires, and then competition among empires begins. In this process, the survival of an empire depends on its power to take over colonies from other empires, and the power of larger empires increases while empires with less power collapse. The weakest empire will be eliminated from the competition, and this algorithm will continue until only one empire remains along with some colonies, which are close to the imperialist country in terms of position. Figure 1 shows the flowchart of the ICA method.

## 2.1 Generating initial empires (Initialization)

The basic goal of the optimization process is to determine values for the variables that minimize or maximize the objective function while satisfying the constraints. In the ICA method, a '$1 * N_{var}$' array called country is introduced as follows:

$$country = [P_1, P_2, \ldots, P_{(N_{var})}] \tag{2.1}$$

where $P_i$ represents a variable of the problem that is interpreted as a socio-political characteristic of a country, such as welfare, culture, religion, and language. The optimal solution to an optimization problem is the one with the maximum power or the minimum cost that is determined by evaluating the objective function ($f$) for variables ($P_1, P_2, \ldots, P_{(N_{var})}$).

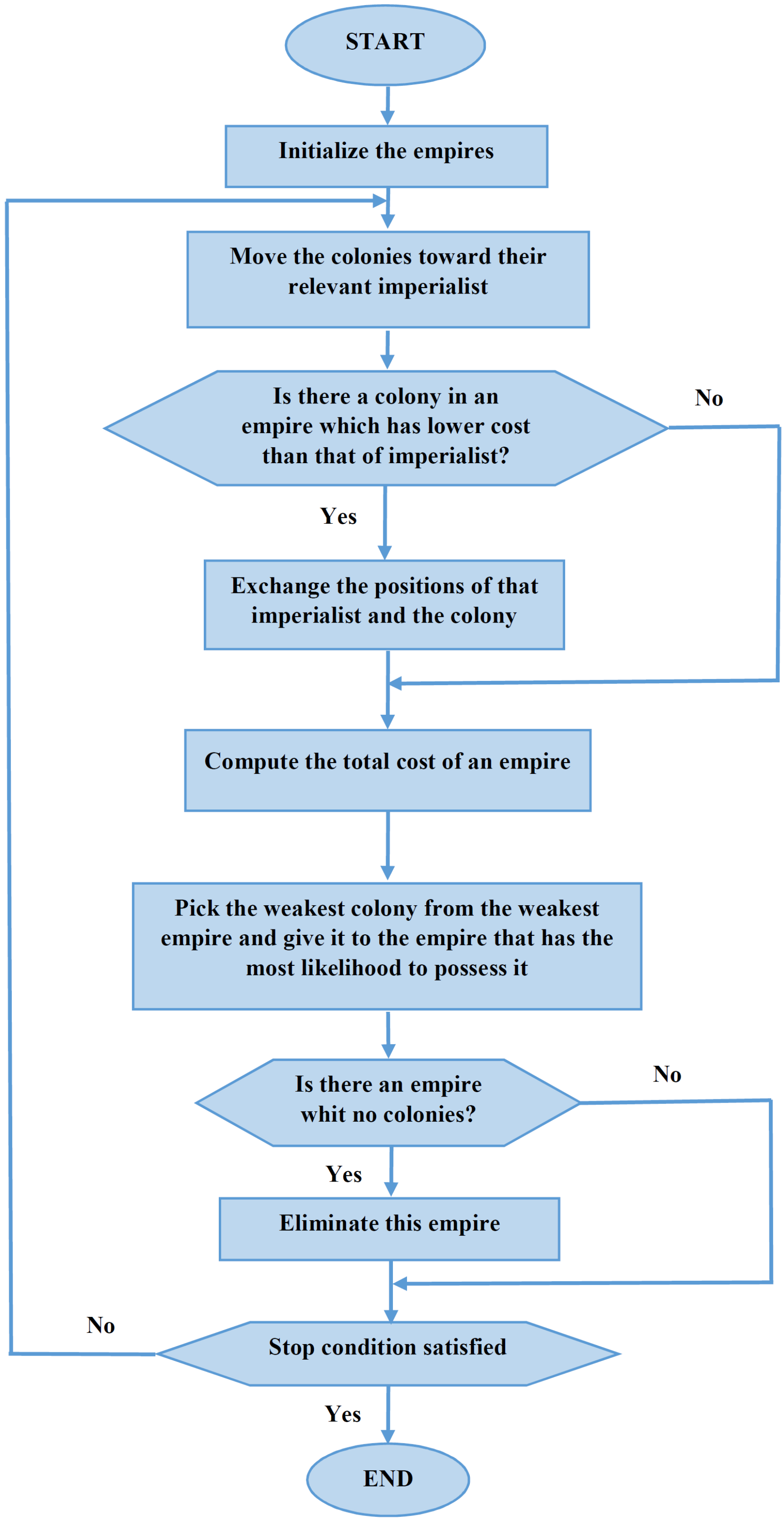


Figure 1: Flowchart of the original ICA

$$cost_i = f(country_i) = f(P_1, P_2, \ldots, P_{(N_{var})}) \tag{2.2}$$

In the first step of the algorithm, the initial population with size $N_{pop}$ is generated. Next, $N_{imp}$ of the best countries, having the lowest cost function values, must be selected as the initial empire's leaders, or imperialists. The remaining countries, $N_{col}$ $(N_{pop} - N_{imp})$, which are considered as colonies, are divided into empires based on imperialist power. The normalized cost of the $n_{th}$ imperialist is given by:

$$C_n = max(c_i) - c_n \tag{2.3}$$

where $c_n$ is the cost of $nth$ imperialist, and $C_n$ is the normalized cost of $nth$ imperialist. An imperialist with a larger cost (i.e. a weaker or low-power imperialist country) has a smaller normalized cost. Therefore, the normalized power of each imperialist is calculated as follows:

$$P_n = \left| \frac{C_n}{\sum_{N_{imp}}^{i=1} C_i} \right| \tag{2.4}$$

The initial colonies are distributed among empires based on their normalized power. Therefore, the initial number of colonies for the nth empire is calculated by:

$$N.C._n = roundP_n.N_{col} \tag{2.5}$$

where $N.C._n$ is the number of initial colonies possessed by the $nth$ imperialist, $N_{col}$ presents the total number of existing colonies in the initial countries, and round is a function that gives the nearest integer of a fractional number. Therefore, each imperialist receives a number of colonies proportional to its power.

## 2.2 Assimilation: Movement of colonies toward their imperialist

After forming initial empires, the assimilation process begins. As shown in Figure 2, countries move toward their imperialists in this process, and this movement is defined by two parameters, $x$ and $\theta$.
Moreover, the colony moves x distance along with d direction toward their corresponding imperialist. Accordingly, $x$ is a random variable with uniform distribution, and this movement can be represented by:

$$x \sim U(0, \beta \times d) \tag{2.6}$$

where d is the distance between the colony and the imperialist, $\beta$ is an assimilation coefficient, and $U$ is the uniform distribution function that takes the two parameters $\beta$ and $d$ to generate a random number between 0 and $\beta \times d$. The deviation parameter is denoted by $\theta$ and follows a uniform distribution:

$$\theta \sim U(-\gamma, \gamma) \tag{2.7}$$

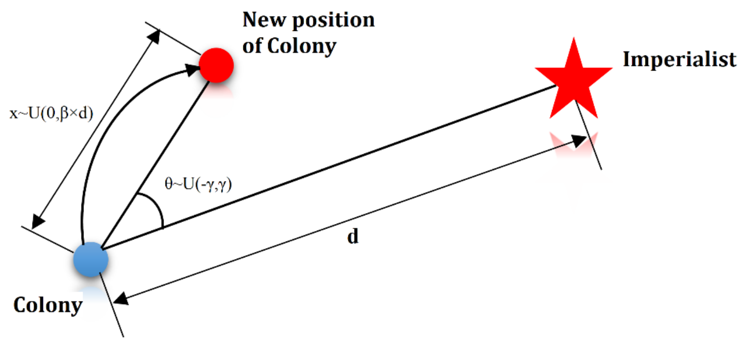


Figure 2: Colonies' movement toward their corresponding imperialist

where $\gamma$ is an arbitrary angle [14]. During the process of assimilation, there is always the possibility that a colony ends up in an even better position than its imperialist. In this case, the colony and the imperialist swap their positions. Subsequently, the algorithm is continued with the new imperialist, and the colonies move toward it.

## 2.3 Revolution

In the real world, all the colonies of an empire aren't attracted by the imperialists in terms of social, cultural, economic and political characteristics, and might be some colonies that resist to be absorbed by imperialists. In fact, the power of some countries might change suddenly due to reformations in their characteristics. This condition in the ICA algorithm is called "revolution". Based on this phenomenon, some colonies' positions in the search space will suddenly change, which increases exploration and prevents the early convergence of countries to local optimum positions. The revolution rate in the ICA illustrates the percentage of colonies that randomly change their position.

## 2.4 Total power of empires

After implementation of the assimilation policy, a total power is assigned for each empire that is equal to the power of the imperialist plus a percentage of the power of the colonies. However, the effect of colonies on the total power is negligible [15, 29]. Therefore, by considering the above both factors, the total power of an empire can be defined as:

$$T.C._n = Cost(imperialist_n) + \zeta mean\{Cost(colonies\, of\, empire_n)\} \tag{2.8}$$

where $T.C._n$ is the total cost of the $nth$ empire, and $\zeta$ is a positive number that has a value between zero and one. In most applications, a small value of 0.1 or 0.15 is considered as an appropriate choice. To model the competition process among empires, the normalized total cost of an empire is calculated as follows:

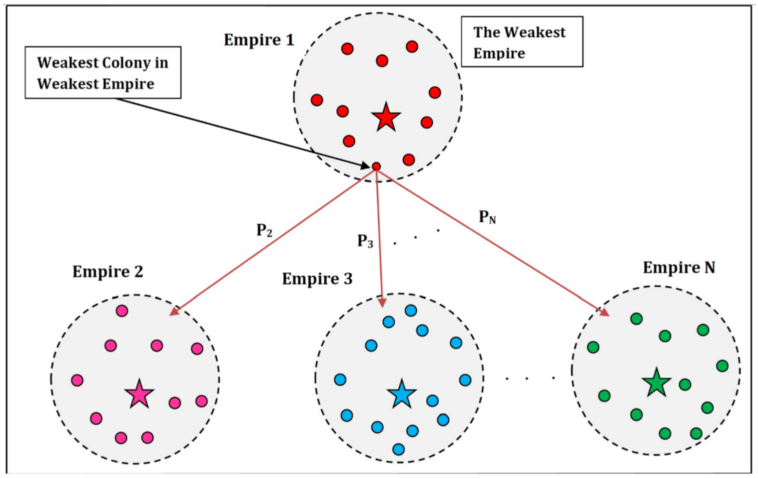


Figure 3: Imperialistic competition

$$N.T.C._n = max\{T.C_i\} - T.C._n \tag{2.9}$$

## 2.5 Imperialistic competition

The imperialistic competition process begins after determining the normalized total cost of each imperialist. This competition in the ICA algorithm works by choosing the weakest colony from the weakest empire and making a competition between all empires to possess this colony Figure 3. The possession probability of each empire, which is proportional to its power, could be computed by:

$$P_{P_n} = \frac{N.T.C._n}{\sum_{N_{imp}}^{i=1} N.T.C_i} \tag{2.10}$$

# 3 Mathematical model of multi-objective problems

Many real-world industrial domains are concerned with large and complex optimization problems involving various conflicting objectives, either to be minimized or maximized, that should be considered simultaneously. Assuming all the objective functions to be minimized, a multi-objective linear problem can be mathematically defined as follows [38, 42]:

$$Minimize \quad \vec{f}(\vec{x}) = [f_1(\vec{x}), f_2(\vec{x}), \ldots, f_k(\vec{x})] \tag{3.1}$$

Subject to:

$$g_i(\vec{x}) \leq 0 \qquad i = 1, 2, \ldots, m$$
$$h_j(\vec{x}) = 0 \qquad j = 1, 2, \ldots, p$$

where $\vec{x} = [x_1, x_2, \ldots, x_n]^T$ represents decision variables vector, $f_i : \mathbb{R}^n \to \mathbb{R}, i = 1, 2, \ldots, k$ are the objective functions, and $g_i, h_j : \mathbb{R}^n \to \mathbb{R}, i = 1, 2, \ldots, m \quad j = 1, 2, \ldots, p$ are inequality and equality constraints, respectively.
The desired solution, known as a Pareto optimal solution, takes the form of "trade-off" solutions between objective functions, with an improvement in one causing a worsening in at least one of the other functions. Accordingly, instead of producing a single solution, a set of optimal solutions called the Pareto optimal set or Pareto optimal solutions is produced [24, 26].
In single-objective optimization problems, there is a single search space, while the search space in multi-objective optimization problems includes the space of design variables and the objective space. Hence, diversity can be defined in both of these spaces. In multi-objective optimization problems, solutions that are close to the true Pareto front are desirable. For a set of objective functions that do not conflict with each other, the Pareto front set would have only one member. While there is a Pareto optimal front for a set of objective functions that conflict with each other. Many multi-objective optimization algorithms use the concept of dominance in their searches. A definition of dominated points is given below.
Solution $S_1$ dominates the other solution $S_2$ if both conditions 1 and 2 are satisfied:

1. In all objectives, solution $S_1$ outperforms or is similar to solution $S_2$.

2. In at least one objective, solution $S_1$ is clearly superior to solution $S_2$.

Based on the above two conditions, if $S_1$ dominates $S_2$, then it is considered a better solution. Among a set of solutions of dominant and non-dominated, those solutions that does not dominate each other are called Pareto front solutions. These non-dominated solutions are joined by a curve defined as the Pareto optimal front.

# 4 Proposed multi-objective imperialistic competitive algorithm

The ICA, as a successful method for single-objective optimization problems, cannot simultaneously manage conflicting objectives in multi-objective optimization problems. In order to extend single-objective ICA to solve multi-objective optimization problems, some modifications should be made to face the fact that the solution of a multi-objective problem is a set of non-dominated solutions. There are modifications to be taken into account when a multi-objective evolutionary algorithm is developed:

1. How to maintain the non-dominated solutions found during the search? It is also desirable to maintain the diversity of the final solutions.

2. How to select countries to be used as imperialists or colonies? In other words, how could the merit of each individual be determined based on all the objective functions?

To deal with the first issue, as has usually been done in some multi-objective evolutionary algorithms, an external archive or repository is used to store the non-dominated solutions searched in the process of solving the optimization problems. In this paper, an external archive is also employed to store non-dominant solutions, and its content will be reported as the final output of the algorithm. Ideally, all non-dominated solutions would be retained in the archive. However, by doing this, its size would quickly increase, especially when dealing with many objectives. To avoid this problem, the size of the archive must be limited. In this paper, to limit the number of non-dominated solutions in the archive, an adaptive elimination approach is employed, which has influences on computational time, convergence, and diversity of solution. This approach would be described in Section 4.3.

Dealing with the second issue is not a trivial task because single-objective ICA does not have the ability to determine the merit of each individual by considering all objectives. An assessment of this merit is necessary to specify imperialistic countries or colonies, through computing the power of imperialistic countries to form empires and to calculate the total power of empires for empire competition. To determine the merit of each individual, fast non-dominated sorting would be implemented in the proposed algorithm according to Section 4.1.1 of this paper in order to rank the solutions and identify imperialists. Moreover, this algorithm takes advantage of an empire formation approach by applying the Sigma method to introduce more convergence. The initial empires are established by utilizing the Sigma method, which is explained in Section 4.1.2. Moreover, when the ICA is applied to high-dimensional or complex multimodal functions, it has the drawback of being trapped in local optimum solutions. As a means of overcoming this drawback and dealing with the diversity of solutions, the Wsum method is used for imperialist competition in order to take over colonies from other empires. This method will be explained in Section 4.2.1.

In summary, the framework of the proposed multi-objective imperialist competitive algorithm is as follows: First, the initial countries are generated randomly. Then, each country is ranked using fast non-dominated sorting technique that takes into account all objectives, and the most powerful countries are saved in archives as imperialists, while the remainders are considered to be colonies of the imperialists. Using the Sigma method, colonies are assigned to imperialists once the type of a country is determined (imperialist or colonial). Following the formation of the initial empires, the process of the assimilation and revolution begins, and colonies move toward their relevant imperialist, similar to the single objective ICA. Moreover, if a colony in the moving process achieves a better position than its imperialist and subsequently dominates it, then the imperialists exchange their positions with that colony. After all the countries are updated, the external archive is updated as well. As the final step of this algorithm, in order to avoid the concentration of solutions in one region of the Pareto front, the WSum method is applied to increase the diversity into the search process. In the following, in order to limit the size of the archive, an adaptive elimination approach is employed. In the end, when the termination condition is satisfied, the archive is returned as the result of the search. Further explanations related to this method are presented below.

## 4.1 Generating initial empires (Initialization)

Initially to form empires, the countries forming the population should be randomly generated. Then the countries with better positions are considered as imperialists, while the rest are called colonies. In order to determine the merit of each individual, the solutions are ranked using the fast non-dominated sorting approach to find the imperialists summarized as follows:

### 4.1.1 Fast non-dominated sorting

Many multi-objective evolutionary algorithms, such as NSGA-II and MOPSO, use fast non-dominated sorting method and crowded distance idea to evaluate individuals. A fast non-dominated sorting strategy categorizes all the solutions into different non-domination ranks based on all the objective functions. By measuring the density of solutions, the crowding distance allows the algorithm to compare two individuals in the same rank and determine which one is better. Non-dominated solutions at this step are assigned to the first non-domination rank called the Pareto front. The procedure will be repeated for all remaining solutions in the population until all solutions become ranked. To do so, the following two parameters would be calculated for each individual:

- $n_p$: domination count, or the number of solutions that dominate the solution $p$.
- $S_p$: a set of solutions in which solution $p$ dominates.

All solutions in the first non-dominated front have $n_p = 0$ (Pareto front). Then, for each solution $p$ with $n_p = 0$, the value of np for each member of set Sp should be revised, and its domination count should be reduced by one. In doing so, if for any member the domination count becomes zero, these members belong to the second non-dominated front. This procedure continues until all solutions to a front are identified. When all countries are ranked using the fast non-dominated sorting method, countries with $n_p = 0$, which are imperialists, are stored in the external archive. The rest of the countries are considered colonies of the imperialists.

Moreover, the Sigma method is used to create initial empires, once the type of country is determined (imperialist or colonial). This method was introduced for the first time by Mostaghim [38] for finding the best local guides for each particle of the population in MOPSO. In MOPSO, selecting the best local guide (leader) for each particle of the population from a set of Pareto-optimal solutions is not trivial. The selection of guides has a significant impact on the convergence and diversity of solutions since it is expected that leadership will direct the search toward better regions and avoid convergence at a single point. In the next section, the Sigma method which is used for the establishment of the initial empires is explained.

### 4.1.2 Sigma method

In the Sigma method, the best local guide for each particle of the population in the objective space is selected based on the sigma distance. For each point with coordinates $(f_{1,i}, f_(2,i))$ in the repository and the swarm, a Sigma vector is calculated. In some cases, the sigma vector may have negative values without compromising its performance because the Euclidean distance between the vectors is calculated. For a problem with two objective functions, the Sigma vector is defined as follows:

$$\sigma = \frac{f_1^2 - f_2^2}{f_1^2 + f_2^2} \tag{4.1}$$

In the general case, for problems with more than two objectives, the Sigma is a vector with $\binom{m}{2}$ elements, where m is the dimension of the objective space. For example, for three coordinates of $f_1$, $f_2$ and $f_3$, it is defined as below.

$$\vec{\sigma} = \begin{pmatrix} f_1^2 - f_2^2 \\ f_2^2 - f_3^2 \\ f_3^2 - f_1^2 \end{pmatrix} /(f_1^2 + f_2^2 + f_3^2) \tag{4.2}$$

In this method, each particle selects the leader candidate among the members of the archive (repository) whose Sigma vector is closer to its own considering the Euclidean distance.

### 4.1.3 Establishment of the initial empires using the Sigma method

Using the basic idea of the Sigma method and by considering the objective space, finding the best imperialist $\mathbf{Imp}_t^{i,g}$ among the archive members for colony $i$ of the population is as below:

In the first step, value $\sigma_j$ for imperialist $j$ in the archive is assigned. In the second step, value $\sigma_i$ is calculated for colony $i$ of the population. Then the distance between the $\sigma_i$ and $\sigma_j$, $\forall j = 1, \ldots, |A|$ is calculated. Finally, imperialist $k$ in the archive which its $\sigma_k$ has the minimum distance to $\sigma_i$ is selected as the imperialist for colony $i$. Therefore, imperialist $\mathbf{Imp}_t^{i,g} = Imp^k$ is the best imperialist for colony $i$. Actually, each colony that has a closer sigma value to the sigma value of the archive member, must select that archive member as its imperialist. The closer meaning in the case of a two-dimensional objective space is the difference between the sigma values. While it means in the m-dimensional objective space, m-Euclidean distance between the sigma values. Figure 4 shows how initial empires can be formed by finding the imperialists among the archive members for

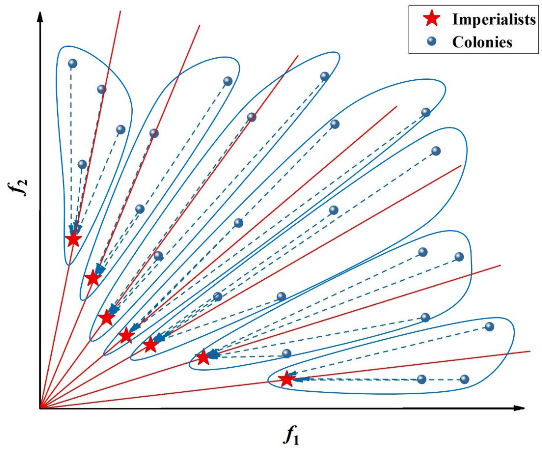


Figure 4: Generating the initial empires using the Sigma method

each colony in a two-dimensional objective space. Moreover, the algorithm of the Sigma method, which is used in this paper to form the initial empires, is shown in Algorithm 1. In this algorithm, the Sigma function calculates the value of $\sigma$ and the Calcdist calculates the Euclidean distance between the inputs. There, $f(Imp)^j$ defines the cost function value of $jth$ element of the archive $A$.

---

**Algorithm 1** The Sigma method for finding the best imperialist $Imp_t^{i,g}$ for colony $i$ of the population

---

**Input:** $A$, $\boldsymbol{Col}^i$
**Output:** $\boldsymbol{Imp}_t^{i,g}$

***Step 1***: Calculate parameter $\sigma$ for the members of A:

**for** $j = 1$ to $|A|$ **do**
  $\sigma_j$ = Sigma($f(Imp)^j$);
**end for**

***Step 2***: Calculate $\sigma_i$ for the colony $i$:
$\sigma_j$ = Sigma($f(Col)^i$);
$dist$ = Calcdist($\sigma_1, \sigma_i$);
**for** $j = 2$ to $|A|$ **do**
  tempdist = Calcdist($\sigma_1, \sigma_i$);
  **if** tempdist $\leq$ dist; **then**
    dist= tempdist
    g=j
  **end if**
**end for**
*END*

---

Through this method, a colony allocation strategy can avoid the disadvantage that the original ICA formula cannot be used to calculate empires' power in multi-objective optimization. It is expected that this method can find solutions with good diversity and convergence.
The algorithm enters the main loop after completing the initial steps and building the initial empires. The process of the assimilation begins, and colonies start to move towards their corresponding imperialists like the single objective ICA. When the assimilation is performed, revolution is another operator of the algorithm that increases the chances of the colonies escaping the local minimum by making random changes to their position depending on the revolution rate. In this paper, the revolution probability was chosen to be 0.1, which means that revolution changes the positions of 10% of the colonies. In addition, the revolution rate was chosen to be 0.05; this rate determines how many variables should revolt in one country. After completing the assimilation and revolution operations, the cost functions of the assimilated and revolted colonies are evaluated. There is always the possibility that a colony in the moving process reaches a better position than its imperialist and dominates it. As a result, the colony and imperialism swap positions, and the colony enters the external archive or repository. Generally, the archive is updated as follows:

- If the new position colony $i$, $Col(i)$, dominates the position of its imperialist $Imp(i)$, clearly $Imp(i)$ will be replaced by $Col(i)$ in the archive.
- In the case that $Imp(i)$ dominates $Col(i)$, $Imp(i)$ will be kept.

## 4.2 Empire Competition

The change in power between empires during the iterations causes a decrease in the power of weaker empires and an increase in the power of stronger empires. In this algorithm, empire competition is modeled by the fact that the most powerful imperialists try to grab colonies from other imperialists, less powerful ones and expand their territory. Competition between empires is actually a process of redistribution of each empire's colonies. However, the method used in basic ICA cannot be used in multi-objective optimization. In fact, in multi-objective problems, it is necessary to establish a quantitative measure to evaluate the solutions and assess the merit of each individual. In imperialist competition, this measure is crucial for determining the power of countries and the probability of empire possession. Therefore, the WSum approach is applied in this study to maintain the diversity of solutions in the competition between empires. In this regard, first, the rank of each country is determined using fast non-dominated sorting by considering all objective functions. In addition, between the countries with the highest and lowest rank (the imperialists in the archive), which are the non-dominant members of the population, the possession probability of each imperialist can be calculated by the following steps.

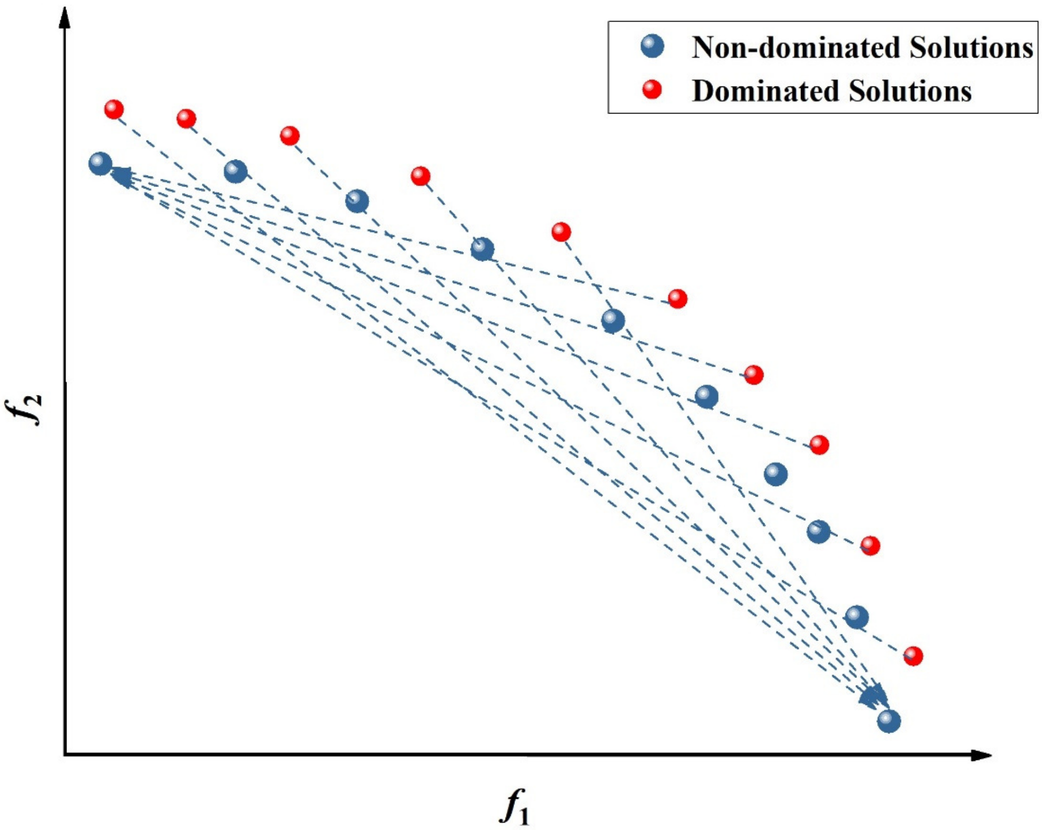


Figure 5: Representation of the WSum method

### 4.2.1 Weighted Sum (WSum)

The weighted sum approach (WSum), developed by Branke and Mostaghim [42] to select the personal best leader for each particle, is a weighted sum of the objective values. In this method, in order to better make diversity, a higher weight is assigned to those criteria in which the particle is already relatively good. In particular, if $f_j(x_i)$ is the $j$-th fitness value of particle $i$, the weighted sum for the particle's personal best is calculated as follows:

$$F = \sum_j \frac{f_j(x_i)}{\sum_k f_k(x_i)} f_j(p_i) \tag{4.3}$$

The personal best that obtains the smallest weighted sum is selected to lead particle $i$. In this method, the selected leader will be the closest to the opposite axis to particle $i$; hence, this method may help maintain a better spread of solutions in the search. Its general behavior (in nonconvex problems) is illustrated in Figure 5.

In this figure, the blue circular points display the leader or personal best candidates in the repository, the red circle points represent the particles in the search space, and the arrows point toward the leader that a particle chooses during its search in a bi-objective optimization problem. In this technique, the most important thing that can be pointed out is that the particles would choose leaders close to the axis but farther from their current position.

### 4.2.2 Empire Competition by using WSum method

In the original ICA, the power of each country is determined based on its objective function, but in the proposed method of this research, the power of each country must be determined based on all objective functions. In this way, the power of each country is mainly considered in relation to its rank. Therefore, the weakest country has a higher rank. After moving the colonies toward their corresponding imperialists, the assimilation and revolution policies would be implemented, and the archive are updated, the process of imperialistic competition begins as follows:
First, the rank of each country is determined using fast non-dominated sorting by considering all objective functions. All countries on the Pareto front which have $n_p = 0$ are stored in the external archive and in fact, are regarded as imperialists. The imperialist countries selected from this set can have a great influence on the convergence and diversity of solutions, and this impact would be greater when the optimization problem has a large number of objective functions. When the rank of all countries is determined, weighted sum $F_c$ for the colonies with the highest rank and the imperialists $c$-th in the archive is calculated by Equation 4.4.

$$F_c = \sum_j \frac{f_j(Col_i)}{\sum_k f_k(Col_i)} f_j(Imp_c) \tag{4.4}$$

In the imperialistic competition process, the weakest colony is assigned to the empire whose weighted sum has the lowest value. Whereas, the power is calculated by Equation 4.5.

$$Power_c = \frac{1}{F_c} \tag{4.5}$$

By calculating the power of all imperialists, the possession probability of each imperialist is defined by:

$$P_c = \left| \frac{Power_c}{\sum_{i=1}^{N_{imp}} Power_i} \right| \tag{4.6}$$

In order to assign the weakest colonies to an empire, the roulette wheel method is used. So there is a competition between the empires to add these colonies to their colonies, and when an imperialist loses all its colonies, it is added as a colony to an empire.

## 4.3 Archive Updating

In this phase, all countries, including imperialists and colonies, are merged into one group, and the fast non-dominated sorting method is applied on it. Once the solutions have been

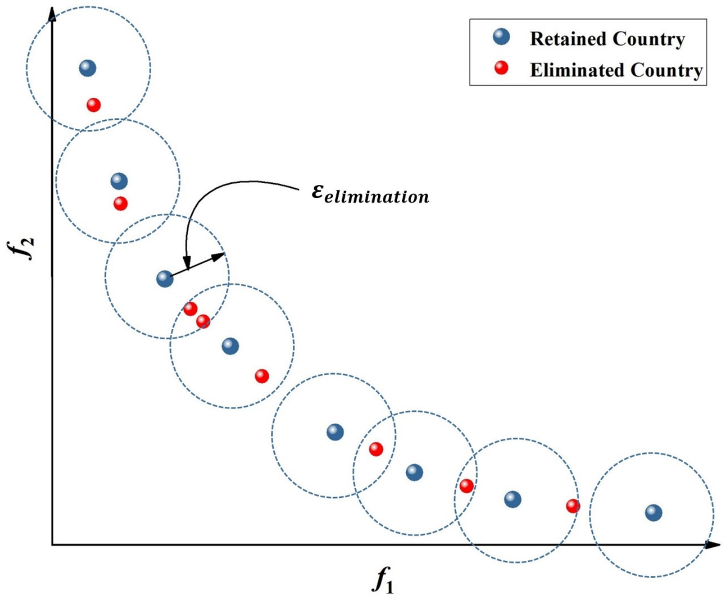


Figure 6: Removing particles located in another particle's radius using $\varepsilon_{elimination}$ approach

ranked, the best non-dominated solutions from these countries, which have a zero rank, are stored in the external archive. The repository, or external archive, maintains the best non-dominated solutions obtained so far during the search. If all non-dominated solutions are kept in the archive, the size of the archive can quickly grow. The archive must be bounded because it would be updated in each iteration, and this updating may become computationally expensive. Therefore, it is necessary to use an additional criterion to decide which non-dominated solutions should be maintained. To limit the number of non-dominated solutions in the archive, an adaptive elimination approach is employed, which was first introduced by Mahmoodabadi [24]. Applying this method affects computation time, convergence, and a variety of solutions. In this technique, each country of the archive has an elimination radius equal to $\varepsilon_{elimination}$, and if the Euclidean distance (in the objective function space) between two countries is less than $\varepsilon_{elimination}$, then one of them will be eliminated. So,Figure 6 shows this approach for a two-objective space. The value of the elimination radius is adaptively calculated according to Equation 4.7.

$$\varepsilon_{elimination} = \frac{t}{\xi \times MaxIt} \tag{4.7}$$

where $\xi$ is a positive constant, t is the current iteration number, and $MaxIt$ is the maximum number of allowable iterations.

As can be seen from Equation 4.7, at the initial iterations, more non-dominated solutions would be kept in the archive (because the elimination radius is small), and this would accelerate the convergence of the algorithm. By increasing the current iteration number, the elimination radius will be larger, and therefore more similar solutions will be eliminated, and this would increase the uniform diversity of non-dominant solutions. The minimum

Euclidean distance between two countries of non-dominated solutions has its maximum value and is equal to $1/\xi$ at the final iteration ($t = MaxIt$).

## 4.4 Implementation of the Proposed Algorithm

The steps of this proposed method include the initialization of solutions, formation of the initial empires, assimilation, revolution of colonies, empire competition, and implementation of an external archive. Among these steps, initialization of solutions, assimilation, and revolution of colonies are the same as those in the single-objective ICA. Previous sections described the steps of forming the initial empires, establishing empire competition, and updating the external archive strategy. Accordingly, in order to employ the proposed method for multi-objective optimization, the following procedure should be followed:

**Step 1**: Set the needed parameters for the algorithm: includeing $n - Pop$ (population size), $n - Var$ (number of decision variables), $VarMin$ (lower bound of variables), $VarMax$ (upper bound of variables), $MaxIt$ (maximum number of iterations), $\beta$ (assimilation coefficient), $p_{Revolution}$ (revolution probability), $\mu$ (revolution rate), and $\zeta$ (positive constant for $\varepsilon_{elimination}$).

**Step 2**: Produce the initial countries, randomly.

**Step 3**: Calculate the objective functions and Sigma value according to Equations (4.1 or 4.2) for every member of the population of the initial countries.

**Step 4**: Create initial empires:

1. Assign the most powerful countries using non-dominated sorting and save them in the archive as the imperialists.
2. Assign other countries to imperialists by using the Sigma method and according to the colonies allocating rules in Section 4.1.2.

**Step 5**: Assimilation and revolution:

1. Move the colonies of an empire toward its imperialist (assimilation).
2. Compute the objective functions of assimilated colonies.
3. Perform revolution probability and revolution rate operations on a new colony.

**Step 6**: If the new colony dominates its own imperialist, then exchange their positions and refresh the external archive.

**Step 7**: Perform the empire competition using the WSum method according to Section 4.2.1.

**Step 8**: Update the external archive according to Section 4.3.

**Step 9**: Perform the process until the maximum number of iterations is reached.

After applying a series of improvements to the algorithm, the pseudocode of the introduced approach is presented as follows.

**Algorithm 2** Pseudocode of the proposed algorithm

**Input:** Set the initial parameters of the proposed Algorithm: $n-Pop$, $n-Var$, $VarMin$, $VarMax$, $MaxIt$, $\beta$, $p_{Revolution}$, $\mu$ and $\zeta$.
**Output:** Pareto front of proposed algorithm.

Create initial countries randomly
**for** $i = 1 : n - Pop$ **do**
Calculate the values of objective functions and Sigma according to Equation (4.1 or 4.2) for each country of the initial populations.
Create empires:
[a] Determine the most powerful countries using non-dominated sorting and reserve them in the archive as the imperialists.
[b] Assign other countries to imperialists using the Sigma method according to the colonies allocating rules in Section 4.1.2.
**end for**
**while** $t \leq MaxIt$ **do**
**for** $i = 1 : n_{Pop}$ **do**
Assimilation, revolution, and imperialist competition process:
[a] Move the colonies of an empire toward its imperialist (assimilation).
[b] Compute the objective functions of assimilated colonies.
[c] Determine revolution probability and revolution rate on a new colony.
[d] Calculate the value of objective functions for each country, and if a new colony dominates its own imperialist, then exchange the new colony and imperialist and refresh the dominant answers in the external archive.
[e] Performing an empire competition using the WSum method according to Section 4.2.1.
[f] Update the external archive according to Section 4.3.
**end for**
**end while**

# 5 Numerical results and experimental validations

This part of the article consists of four sub-sections: The first is on standard test functions, which represent benchmark problems for bi-objective and tri-objective optimization. These standard test functions should provide enough difficulty to challenge the algorithm in searching for the Pareto optimal solutions. The second section is about four performance metrics which represent the convergence and diversity of the final solutions. The third section includes comparative algorithms and simulation settings. These algorithms, which include NSGA-II, MOEA/D, SPEA2, PESA-II, GrEA, and KnEA, have provided acceptable results in solving these types of problems. The related parameter settings for these algorithms are also given in this section. The last section is simulations and discussions, which compare the results of the proposed algorithm with other algorithms in the form of graphs and charts. The interpretations of these charts and graphs are also given in this section.

## 5.1 Standard test functions

In order to verify the performance of the algorithm proposed in this paper, 12 benchmark functions are employed. These test functions have various features such as multimodality, convexity, discontinuity, and non-uniformity, which may limit the ability of the algorithm to control the convergence and diversity in multi-objective problems. The mathematical expressions and the admissible ranges of their variables for all benchmarks are summarized in Tables 1 and 2. In order to achieve reliable results, for all selected benchmark functions, 30 independent runs of each algorithm are performed, and the average of the best results is shown. In fact, test functions can be divided into two categories:
ZDT test functions. These benchmarks, represented in Table 1, were developed by Zitzler et al [26] which are the most widely employed suite of benchmarks in standard bi-objective optimization problems.
DTLZ test functions. These benchmarks, represented in Table 2, were developed by Deb et al. [40] which are employed suite of benchmarks in standard tri-objective optimization problems.

## 5.2 Performance metrics

To assess the performance of a multi-objective optimizer, two basic issues should be considered. First, the ability to reach the optimal set solutions, and second, the uniform distribution along the Pareto front. Here, in order to evaluate the convergence and distribution of solutions, four metrics, i.e. generational distance ($GD$), inverted generational distance ($IGD$), spacing ($S$), and spread ($\Delta$) are adopted. The descriptions of these four indicators is briefly as follows.

- Generational distance ($GD$): This metric, which was defined by Van Veldhuizen and Lamont [22], refers to the distance between the non-dominated solution members obtained by the algorithm and the true Pareto front. It is clear that the algorithms with the lowest $GD$ have the best convergence to the Pareto optimal front. This metric is defined as:

$$GD(PF, PF^*) = \frac{\sqrt{\sum_{i=1}^{n_{nd}} (min \parallel PF_i, PF_i^* \parallel)^2}}{n_{nd}} \tag{5.1}$$

  where $n_{PF}$ is the number of members in the true Pareto front.

Table 1: The mathematical expressions of the $ZDT$ bi-objective test function suite.

| Function name | Objective functions | Dimension $(n-Var)$ | Bounds $[VarMin, VarMax]$ |
|---|---|---|---|
| ZDT1 | $f_1(x)=x_1$ <br> $f_2(x)=g(x)\left(1-\sqrt{\frac{f_1(x)}{g(x)}}\right)$ <br> $g(x)=1+\frac{9(\sum_{i=2}^{n}x_i)}{n-1}$ | n=30 | $x_i\in[0,1]$ |
| ZDT2 | $f_1(x)=x_1$ <br> $f_2(x)=g(x)\left(1-\frac{f_1(x)}{g(x)}\right)^2$ <br> $g(x)=1+\frac{9(\sum_{i=2}^{n}x_i)}{n-1}$ | n=30 | $x_i\in[0,1]$ |
| ZDT3 | $f_1(x)=x_1$ <br> $f_2(x)=g(x)\left[1-\sqrt{\frac{f_1(x)}{g(x)}}-\frac{f_1(x)}{g(x)}\sin(10\pi x_i)\right]$ <br> $g(x)=1+\frac{9(\sum_{i=2}^{n}x_i)}{n-1}$ | n=30 | $x_i\in[0,1]$ |
| ZDT4 | $f_1(x)=x_1$ <br> $f_2(x)=g(x)\left(1-\sqrt{\frac{f_1(x)}{g(x)}}\right)$ <br> $g(x)=1+10(n-1)+\sum_{i=2}^{n}\left[x_i^2-10\cos(4\pi x_i)\right]$ | n=10 | $x_1\in[0,1]$ and $x_i\in[-5,5]$ $i=2,\ldots,n$ |
| ZDT5 | $f_1(x)=x_1$ <br> $f_2(x)=g(x)\left(\frac{1}{f_1(X)}\right)$ <br> $g(x)=\sum_{i=2}^{n}\nu(u(x_i))$ <br> $\nu(u(x_i))=\begin{cases}2+u(x_i), & \text{if } u(x_i)<5\\ 1, & \text{if } u(x_i)=5\end{cases}$ | n=11 | $x_1\in[0,1]^{30}$ and $x_i\in\{0,1\}^5$ $i=2,\ldots,n$ |
| ZDT6 | $f_1(x)=1-\exp(-4x_1)\sin^6(6\pi x_1)$ <br> $f_2(x)=g(x)\left[1-\left(\frac{f_1(x)}{g(x)}\right)^2\right]$ <br> $g(x)=1+9\left(\frac{\sum_{i=2}^{n}x_i}{n-1}\right)^{0.25}$ | n=10 | $x_i\in[0,1]$ |

Table 2: The mathematical expressions of the $DTLZ$ tri-objective test function suite.

| Function name | Objective functions | Dimension (n-Var) | Bounds [VarMin,VarMax] |
|---|---|---|---|
| DTLZ1 | $f_1=(1/2)x_1x_2(1+g)$ <br> $f_2=(1/2)x_1(1-x_2)(1+g)$ <br> $f_3=(1/2)(1-x_1)(1+g)$ <br> $g=100[10+\sum_{i=3}^{n}((x_i-0.5)^2-\cos(20\pi(x_i-0.5)))]$ | 12 | $[0,1]$ |
| DTLZ2 | $f_1=(1+g)\cos(x_1(\pi/2))\cos(x_2(\pi/2))$ <br> $f_2=(1+g)\cos(x_1(\pi/2))\sin(x_2(\pi/2))$ <br> $f_3=(1+g)\sin(x_1(\pi/2))$ <br> $g=\sum_{i=3}^{n}(x_i-0.5)^2$ | 12 | $[0,1]$ |
| DTLZ3 | $f_1=(1+g)\cos(x_1(\pi/2))\cos(x_2(\pi/2))$ <br> $f_2=(1+g)\cos(x_1(\pi/2))\sin(x_2(\pi/2))$ <br> $f_3=(1+g)\sin(x_1(\pi/2))$ <br> $g=100[10+\sum_{i=3}^{n}((x_i-0.5)^2-\cos(20\pi(x_i-0.5)))]$ | 12 | $[0,1]$ |
| DTLZ4 | $f_1=(1+g)\cos(x_1^{\alpha}(\pi/2))\cos(x_2^{\alpha}(\pi/2))$ <br> $f_2=(1+g)\cos(x_1^{\alpha}(\pi/2))\sin(x_2^{\alpha}(\pi/2))$ <br> $f_3=(1+g)\sin(x_1^{\alpha}(\pi/2))$ <br> $g=\sum_{i=3}^{n}(x_i-0.5)^2$, where $\alpha=100$ | 12 | $[0,1]$ |
| DTLZ6 | $f_1=(1+g)\cos((\pi/2)\theta_1)\cos((\pi/2)\theta_2)$ <br> $f_2=(1+g)\cos((\pi/2)\theta_1)\sin((\pi/2)\theta_2)$ <br> $f_3=(1+g)\sin((\pi/2)\theta_1)$ <br> $\theta_1=\frac{\pi}{2}x_1$ , $\theta_2=(\pi/4(1+g))(1+2gx_2)$ <br> $g=\sum_{x_i}(x_i)^{0.1}$ | 12 | $[0,1]$ |
| DTLZ7 | $f_1=x_1$ <br> $f_2=x_2$ <br> $f_3=(1+g)(3-\sum_{i=1}^{2}\left(\frac{f_i}{1+g}(1+\sin(3\pi f_i))\right)$ <br> $g=1+\frac{9}{22}\sum_{i=3}^{n}x_i$ | 22 | $[0,1]$ |

- Spacing ($S$): This metric, which was defined by [16], refers to the spread of the non-dominated solutions and indicates the uniform distribution of them along the obtained Pareto front. This metric can be calculated by:

$$S(PF, PF^*) = \sqrt{\frac{1}{n_{nd}-1}\sum_{i=1}^{n_{nd}}(d_i - \bar{d}_i)^2} \tag{5.2}$$

where $d_i = min_j(|f_1^i(x) - f_1^j(x)| + |f_2^i(x) - f_2^j(x)|)$, $i, j = 1, \ldots, n_{nd}$, and $\bar{d}$ is the mean of all $d_i$. It should be noted that $S = 0$ illustrates the best uniform distribution in the obtained set of non-dominated solutions and indicates that all members are evenly spaced apart.

- Spread ($\Delta$): This metric shows the distribution of the solutions and the diversity in the non-dominated solutions obtained through the optimization processes. This metric, which was defined by Deb [35], is expressed as follows:

$$\Delta(PF, PF^*) = \frac{d_f + d_t + \sum_{i=1}^{n_{nd}} |d_i - \bar{d}|}{d_f + d_t + (n_{nd} - 1)\bar{d}} \tag{5.3}$$

The parameters $d_f$ and dl are the Euclidean distance between the extreme solutions in the true Pareto front and the boundary solutions in the obtained non-dominated front. $d_i$ is the Euclidean distance between consecutive solutions in the obtained set of the non-dominated front, and $\bar{d}$ is the mean of these distances. In general, the value of $\Delta$ is always greater than zero, and if it is equal to zero, excellent conditions occur. Its lower value means the better distribution and expansion of solutions, and its zero value indicates that extreme solutions of the true Pareto front have been found, and for all non-dominated points ($d_i = \bar{d}$).

## 5.3 Comparative algorithms and simulating settings

In this study, to investigate the efficiency and effectiveness of the proposed algorithm, the results would be compared with six popular multi-objective algorithms, including NSGA-II, MOEA/D, SPEA2, PESA-II, GrEA, and KnEA. This comparison is based on the 4 performance metrics and 12 benchmark functions described in the previous sub-sections. The proposed algorithm is coded in the programming section of MATLAB R2018a 64-bit (Win64) software and simulation are performed in Windows 10, Intel®Core(TM) i7-7700HQ CPU 2.80GHz with a 16.00 GB RAM memory. In order to achieve reliable results, all the benchmark functions are tested with 30 independent runs. The mean and standard deviation values of the performance metrics corresponding to each test function are shown in Tables 3 $\sim$ 6. In all these multi-objective comparative algorithms, the related parameter settings are the same as in their corresponding references [32, 33, Khanali[23, 25, 27, 18]. The initial population size of the proposed algorithm is set to 200. The maximum number of iterations ($MaxIt$) for bi-objective benchmark

Table 3: Result of generational distance ($GD$) on the $ZDT$ bi-objective and $DTLZ$ tri-objective test function suites.

| Benchmark Functions | | NSGA-II | MOEA/D | SPEA2 | PESA-II | GrEA | KnEA | Proposed Algorithm |
|---|---|---|---|---|---|---|---|---|
| ZDT1 | Mean | 1.466E-4 | 2.569E-4 | 1.898E-4 | 3.631E-4 | 4.684E-4 | 1.594E-4 | 1.301E-4 |
| | SD | 3.532E-5 | 6.461E-5 | 4.702E-5 | 9.413E-4 | 1.772E-3 | 5.091E-4 | 1.601E-5 |
| ZDT2 | Mean | 1.295E-4 | 3.782E-4 | 1.457E-4 | 4.704E-4 | 1.232E-4 | 6.587E-5 | 3.735E-4 |
| | SD | 4.301E-5 | 2.962E-4 | 4.561E-5 | 5.232E-3 | 3.981E-5 | 3.652E-5 | 2.051E-4 |
| ZDT3 | Mean | 8.606E-5 | 1.062E-3 | 1.817E-4 | 1.742E-3 | 1.409E-4 | 1.195E-4 | 1.086E-4 |
| | SD | 2.571E-5 | 1.682E-3 | 5.933E-4 | 6.992E-3 | 2.071E-4 | 2.692E-4 | 2.522e-5 |
| ZDT4 | Mean | 2.391E-4 | 1.574E-3 | 2.796E-4 | 1.308E-3 | 3.238E-2 | 9.596E-3 | 1.197E-4 |
| | SD | 1.292E-4 | 1.421E-3 | 1.293E-4 | 7.182E-3 | 5.103E-2 | 2.341E-2 | 1.251E-4 |
| ZDT5 | Mean | 3.533E-2 | 1.321E-1 | 5.068E-2 | 1.976E-2 | 7.035E-2 | 2.727E-2 | 1.578E-2 |
| | SD | 1.351E-2 | 6.932E-3 | 2.061E-2 | 2.672E-2 | 2.561E-2 | 1.550E-2 | 1.462E-2 |
| ZDT6 | Mean | 1.318E-4 | 5.536E-4 | 1.933E-4 | 8.096E-4 | 1.316E-3 | 7.919E-4 | 1.152E-4 |
| | SD | 9.692E-5 | 1.591E-4 | 1.453E-4 | 1.354E-4 | 4.531E-3 | 6.812E-4 | 1.235E-4 |
| DTLZ1 | Mean | 9.162E-4 | 3.993E-4 | 2.878E-3 | 8.028E-2 | 1.569E-1 | 5.438E-2 | 2.979E-4 |
| | SD | 3.342E-3 | 1.663E-4 | 1.513E-2 | 1.903E-1 | 2.922E-1 | 1.184E-1 | 1.125e-4 |
| DTLZ2 | Mean | 7.606E-4 | 3.689E-4 | 6.434E-4 | 8.063E-4 | 3.389E-4 | 3.674E-4 | 2.538E-4 |
| | SD | 7.154E-5 | 4.873E-6 | 4.851E-5 | 8.312E-5 | 2.014E-5 | 3.075E-5 | 3.851E-6 |
| DTLZ3 | Mean | 1.370E-1 | 1.060E-1 | 1.454E-1 | 1.221E-1 | 1.750E-1 | 4.777E-2 | 1.030E-1 |
| | SD | 2.463E-1 | 1.542E-1 | 2.164E-1 | 2.644E-1 | 2.092E-1 | 5.963E-2 | 1.331E-1 |
| DTLZ4 | Mean | 7.385E-4 | 2.983E-4 | 6.931E-4 | 8.196E-4 | 3.439E-4 | 3.512E-4 | 2.424E-4 |
| | SD | 6.004E-5 | 1.014E-4 | 1.692E-4 | 7.423E-5 | 5.953E-5 | 1.972E-5 | 2.351E-5 |
| DTLZ6 | Mean | 3.437E-6 | 5.244E-4 | 3.376E-6 | 3.891E-6 | 3.273E-6 | 3.129E-6 | 2.122E-6 |
| | SD | 1.183E-7 | 7.234E-3 | 1.192E-7 | 1.231E-7 | 3.063E-7 | 2.064E-7 | 1.023E-7 |
| DTLZ7 | Mean | 1.726E-3 | 2.419E-3 | 1.343E-3 | 1.712E-3 | 4.948E-4 | 8.937E-4 | 1.041E-3 |
| | SD | 1.991E-4 | 5.634E-4 | 1.253E-4 | 3.282E-4 | 8.031E-5 | 1.193E-4 | 2.034E-4 |

functions is 100, and the maximum number of evaluations is 10000, whereas, for tri-objective benchmark functions, these values are 300 and 50000, respectively. Furthermore, assimilation coefficient $\beta = 2$, revolution probability $p_{Revolution} = 0.1$, revolution rate $\mu = 0.05$, and the positive constant for $\varepsilon_{elimination}$ of $\xi = 100$ . When calculating a benchmark function, to have a fair comparison with the results of different algorithms, the initial population size, the maximum number of iterations and the maximum number of evaluations of all comparison algorithms are same as those of the proposed algorithm.

## 5.4 Experimental Results and Discussion

As mentioned in previous sections, the proposed algorithm in this paper uses the Sigma method in empire formation to improve the convergence, and the $WSum$ method in the competition between empires to increase the diversity of solutions. Therefore, it is expected to be computationally more efficient or comparable to the most efficient algorithms. In order to assess the effectiveness of the new algorithm, four performance metrics of $GD$, $IGD$, $S$, and $\Delta$ are used. The results are compared with six multi-objective algorithms: $NSGA-II$, $MOEA/D$, $SPEA2$, $PESA-II$, $GrEA$, and $KnEA$. Tables 3, 4, 5, and 6 show the mean and standard deviation of the $GD$, $IGD$, $S$, and $\Delta$ values averaged over 30 independent runs for the seven compared $MOEA$s, with the best mean highlighted among the compared algorithms. Moreover, in order to illustrate the convergence and distribution of the solutions on the obtained Pareto fronts by the proposed algorithm, the resultant Pareto front generated by three algorithms in solving the twelve test functions in the best run are shown in Figures 7 $\sim$ 18. To summarize, as can be seen from tables and figures, non-dominated solutions found by the proposed algorithm are very near the true Pareto front, and they are superior in the majority of the standard bi-objective and tri-objective test functions.

Table 4: Result of inverted generational distance ($IGD$) on the $ZDT$ bi-objective and $DTLZ$ tri-objective test function suites.

| Benchmark Functions | | NSGA-II | MOEA/D | SPEA2 | PESA-II | GrEA | KnEA | Proposed Algorithm |
|---|---|---|---|---|---|---|---|---|
| **ZDT1** | Mean | 4.793E-3 | 1.824E-2 | 3.509E-3 | 2.304E-2 | 5.255E-3 | 3.416E-2 | 1.978E-3 |
| | SD | 7.421E-3 | 2.154E-2 | 6.391E-4 | 2.633E-2 | 4.331E-4 | 3.963E-2 | 2.042E-3 |
| **ZDT2** | Mean | 1.565E-2 | 4.487E-2 | 1.257E-2 | 9.932E-1 | 1.444E-2 | 2.434E-2 | 1.636E-2 |
| | SD | 2.463E-2 | 8.523E-2 | 1.752E-2 | 1.122E-2 | 1.832E-2 | 2.374E-2 | 1.073E-2 |
| **ZDT3** | Mean | 4.074E-2 | 2.492E-2 | 1.629E-2 | 1.371E-1 | 2.184E-2 | 2.772E-2 | 1.081E-2 |
| | SD | 6.284E-2 | 2.263E-2 | 2.544E-2 | 9.154E-2 | 3.204E-2 | 4.013E-2 | 1.872E-2 |
| **ZDT4** | Mean | 6.212E-3 | 2.862E-2 | 4.597E-3 | 8.627E-3 | 3.007E-2 | 4.742E-2 | 1.736E-2 |
| | SD | 1.232E-2 | 2.312E-2 | 1.573E-3 | 2.273E-3 | 3.632E-2 | 5.462E-2 | 1.144E-2 |
| **ZDT5** | Mean | 4.229E-1 | 7.859E+0 | 3.660E-1 | 5.180E-1 | 2.074E+0 | 5.850E+0 | 3.076E-1 |
| | SD | 1.123E-1 | 2.056E-1 | 1.213E-1 | 2.782E-1 | 2.584E-1 | 1.553E+0 | 1.485E-1 |
| **ZDT6** | Mean | 2.566E-3 | 6.488E-3 | 3.091E-3 | 4.329E-3 | 4.450E-3 | 3.244E-3 | 5.275E-3 |
| | SD | 6.561E-4 | 1.767E-3 | 1.342E-3 | 6.114E-4 | 5.861E-4 | 6.046E-4 | 6.863E-3 |
| **DTLZ1** | Mean | 2.283E-2 | 1.503E-2 | 1.514E-2 | 1.938E-2 | 5.623E-2 | 3.188E-2 | 2.167E-2 |
| | SD | 1.693E-2 | 9.564E-4 | 9.693E-4 | 1.343E-3 | 4.462E-2 | 2.463E-2 | 1.294E-2 |
| **DTLZ2** | Mean | 4.891E-2 | 3.649E-2 | 3.792E-2 | 4.520E-2 | 5.448E-2 | 4.415E-2 | 3.376E-2 |
| | SD | 1.684E-3 | 7.293E-5 | 3.164E-4 | 8.802E-4 | 7.391E-4 | 1.182E-3 | 1.034E-3 |
| **DTLZ3** | Mean | 6.959E-1 | 5.894E-1 | 6.404E-1 | 4.004E-1 | 1.112E+0 | 4.143E-1 | 4.043E-1 |
| | SD | 7.661E-1 | 1.452E+0 | 7.332E-1 | 6.292E-1 | 1.052E+0 | 4.873E-1 | 5.175E-1 |
| **DTLZ4** | Mean | 4.789E-2 | 3.242E-1 | 8.004E-2 | 4.572E-2 | 8.771E-2 | 4.394E-2 | 2.483E-2 |
| | SD | 1.162E-3 | 2.974E-1 | 1.404E-1 | 8.061E-4 | 1.243E-1 | 1.414E-3 | 1.182E-2 |
| **DTLZ6** | Mean | 2.897E-3 | 2.241E-2 | 2.042E-3 | 7.550E-3 | 2.209E-2 | 2.668E-3 | 6.929E-3 |
| | SD | 1.074E-4 | 3.293E-4 | 1.083E-5 | 1.114E-3 | 2.714E-4 | 4.386E-4 | 7.483E-4 |
| **DTLZ7** | Mean | 5.374E-2 | 1.729E-1 | 4.168E-2 | 8.965E-2 | 6.773E-2 | 4.508E-2 | 3.022E-2 |
| | SD | 2.753E-3 | 2.104E-1 | 9.402E-4 | 1.252E-1 | 4.182E-3 | 1.823E-3 | 1.242E-3 |

Table 5: Result of spacing ($S$) on the $ZDT$ bi-objective and $DTLZ$ tri-objective test function suites.

| Benchmark Functions | | NSGA-II | MOEA/D | SPEA2 | PESA-II | GrEA | KnEA | Proposed Algorithm |
|---|---|---|---|---|---|---|---|---|
| **ZDT1** | Mean | 3.393E-3 | 3.918E-3 | 1.845E-3 | 8.430E-3 | 9.290E-3 | 6.733E-3 | 1.633E-3 |
| | SD | 4.013E-4 | 9.741E-4 | 1.782E-4 | 1.281E-2 | 2.552E-2 | 6.614E-3 | 1.153E-3 |
| **ZDT2** | Mean | 4.153E-3 | 4.397E-3 | 3.783E-3 | 8.104E-3 | 3.830E-3 | 6.567E-3 | 3.162E-3 |
| | SD | 2.752E-3 | 1.874E-3 | 2.513E-3 | 1.713E-2 | 9.954E-4 | 3.205E-3 | 2.724E-3 |
| **ZDT3** | Mean | 4.447E-3 | 1.213E-2 | 2.157E-3 | 5.651E-2 | 7.533E-3 | 7.078E-3 | 4.322E-3 |
| | SD | 6.174E-3 | 2.742E-3 | 4.165E-4 | 9.683E-3 | 2.723E-3 | 5.334E-3 | 8.843E-4 |
| **ZDT4** | Mean | 3.318E-3 | 5.326E-3 | 1.950E-3 | 1.948E-2 | 4.024E-1 | 1.335E-1 | 1.240E-3 |
| | SD | 3.302E-4 | 1.243E-3 | 2.692E-4 | 1.024E-1 | 6.573E-1 | 3.261E-1 | 2.032E-4 |
| **ZDT5** | Mean | 5.982E-3 | 5.980E-1 | 2.393E-3 | 6.758E-2 | 1.088E-1 | 2.272E-1 | 4.818E-2 |
| | SD | 2.004E-2 | 2.632E-2 | 1.316E-2 | 4.542E-2 | 3.832E-2 | 9.174E-2 | 1.861E-2 |
| **ZDT6** | Mean | 2.801E-3 | 2.648E-3 | 1.730E-3 | 6.854E-2 | 1.973E-2 | 5.537E-3 | 3.928E-4 |
| | SD | 8.023E-4 | 7.304E-4 | 1.213E-3 | 1.153E-1 | 6.361E-2 | 4.913E-4 | 1.264E-4 |
| **DTLZ1** | Mean | 1.626E-2 | 3.790E-3 | 3.766E-2 | 1.101E+0 | 2.144E+0 | 7.399E-1 | 1.597E-3 |
| | SD | 4.253E-3 | 1.362E-3 | 1.912E-1 | 2.614E+0 | 4.083E+0 | 1.663E+0 | 2.425E-3 |
| **DTLZ2** | Mean | 3.996E-2 | 3.772E-2 | 1.637E-2 | 3.923E-2 | 4.365E-2 | 5.023E-2 | 5.820E-3 |
| | SD | 2.451E-3 | 2.874E-4 | 9.024E-4 | 2.604E-3 | 1.982E-3 | 2.205E-3 | 2.303E-3 |
| **DTLZ3** | Mean | 9.319E-1 | 1.321E-1 | 1.244E+0 | 1.131E+0 | 4.657E-1 | 8.845E-2 | 4.315E-1 |
| | SD | 3.415E+0 | 1.282E-1 | 3.114E+0 | 3.592E+0 | 1.514E+0 | 5.226E-2 | 5.831E-1 |
| **DTLZ4** | Mean | 3.985E-2 | 2.069E-2 | 1.505E-2 | 4.027E-2 | 4.091E-2 | 5.108E-2 | 5.016E-3 |
| | SD | 1.964E-3 | 1.641E-2 | 4.304E-3 | 2.083E-3 | 1.062E-2 | 2.342e-3 | 2.852E-4 |
| **DTLZ6** | Mean | 5.554E-3 | 5.459E-2 | 2.302E-3 | 3.207E-3 | 7.926E-3 | 5.753E-3 | 2.466E-3 |
| | SD | 3.214E-4 | 7.053E-3 | 1.353E-4 | 2.662E-4 | 9.415E-5 | 9.414E-4 | 1.934E-3 |
| **DTLZ7** | Mean | 4.656E-2 | 1.231E-1 | 2.006E-2 | 4.273E-2 | 3.918E-2 | 4.864E-2 | 1.134E-2 |
| | SD | 4.542E-3 | 3.162E-2 | 1.491E-3 | 6.284E-3 | 6.003E-3 | 3.183E-3 | 2.652E-3 |

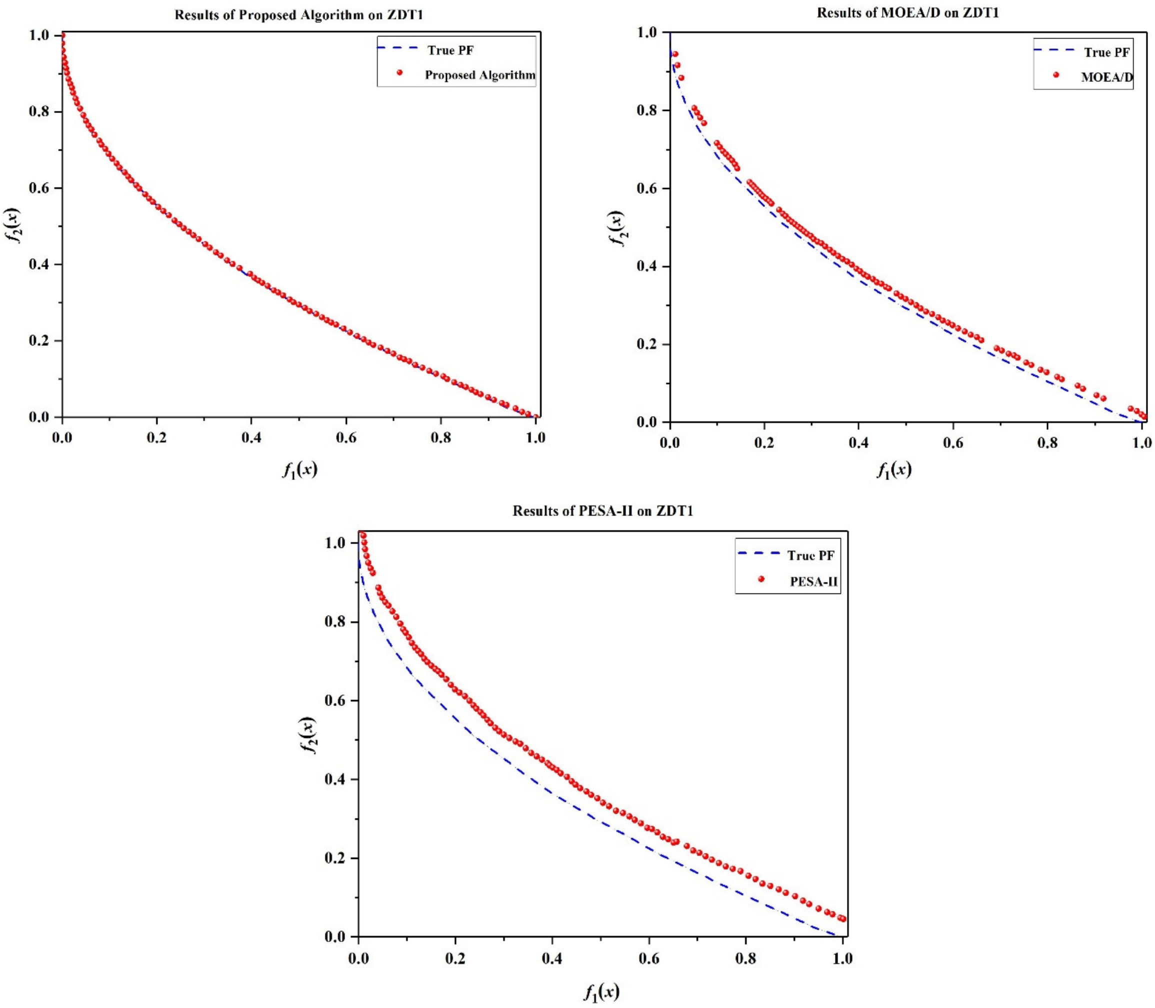


Figure 7: Pareto frontiers of the $ZDT1$ benchmark function produced by $PESA-II$, $MOEA/D$, and the Proposed Algorithm

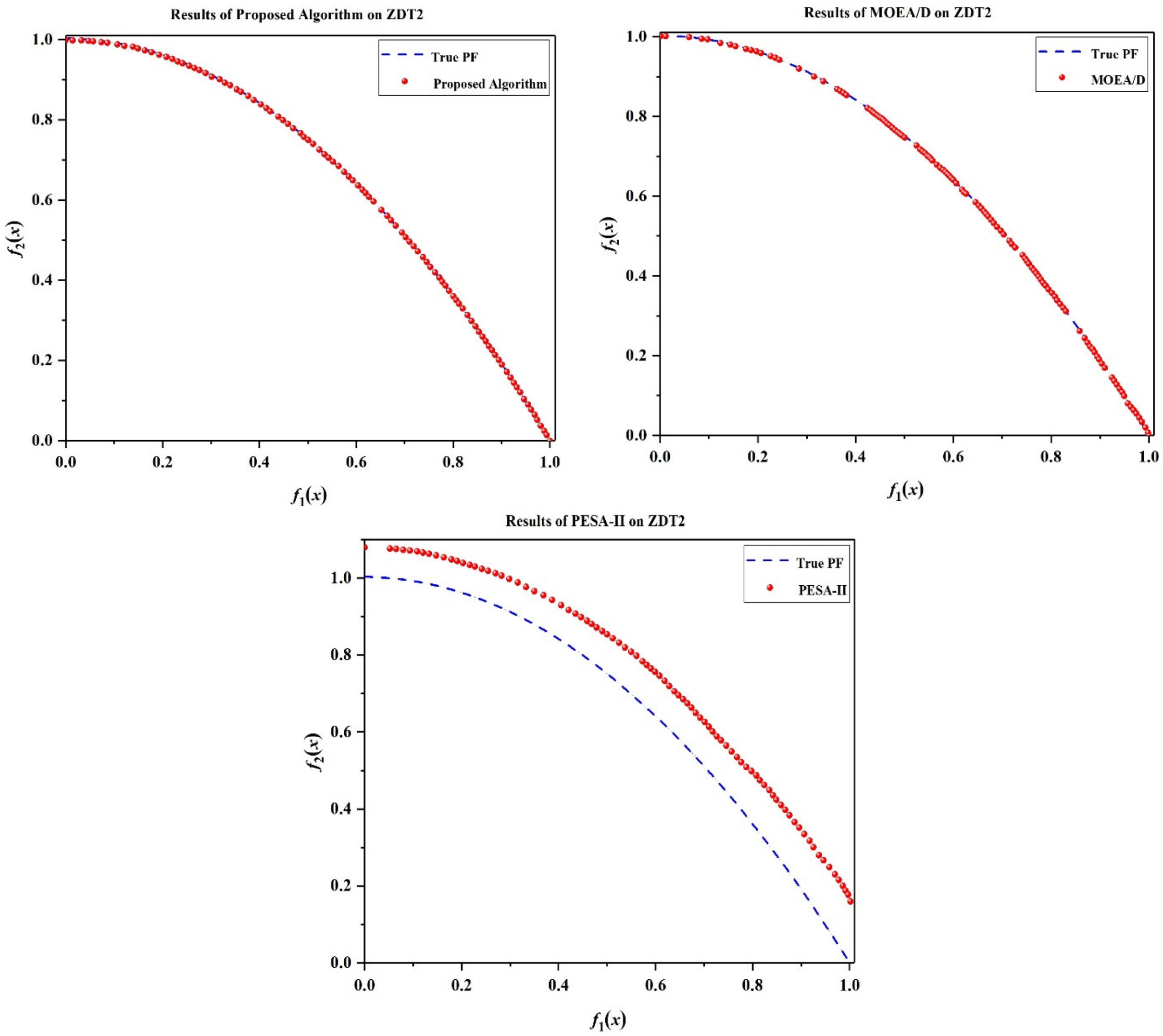


Figure 8: Pareto frontiers of the $ZDT2$ benchmark function produced by $PESA - II$, $MOEA/D$, and the Proposed Algorithm.

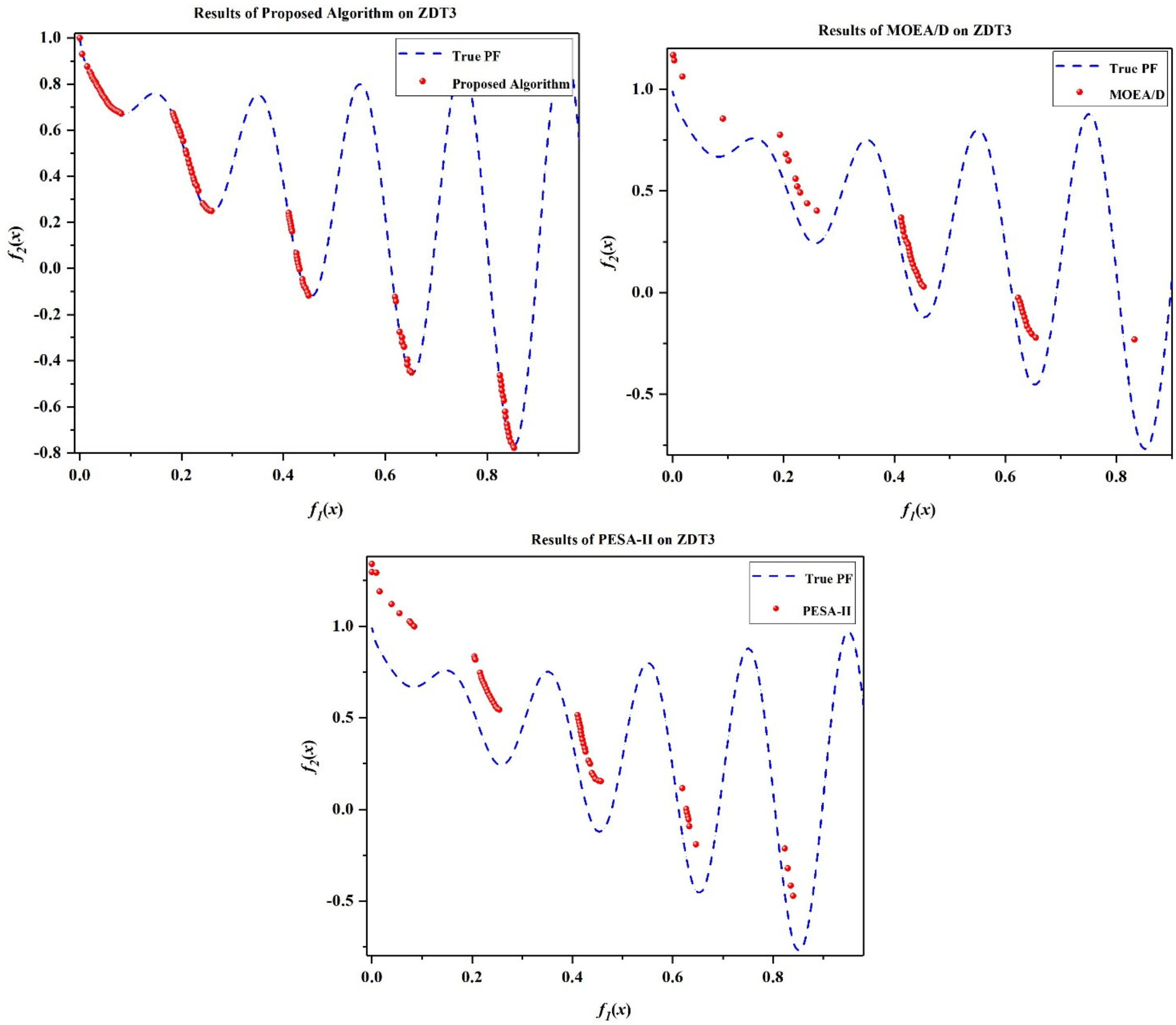


Figure 9: Pareto frontiers of the $ZDT3$ benchmark function produced by $PESA - II$, $MOEA/D$, and the Proposed Algorithm

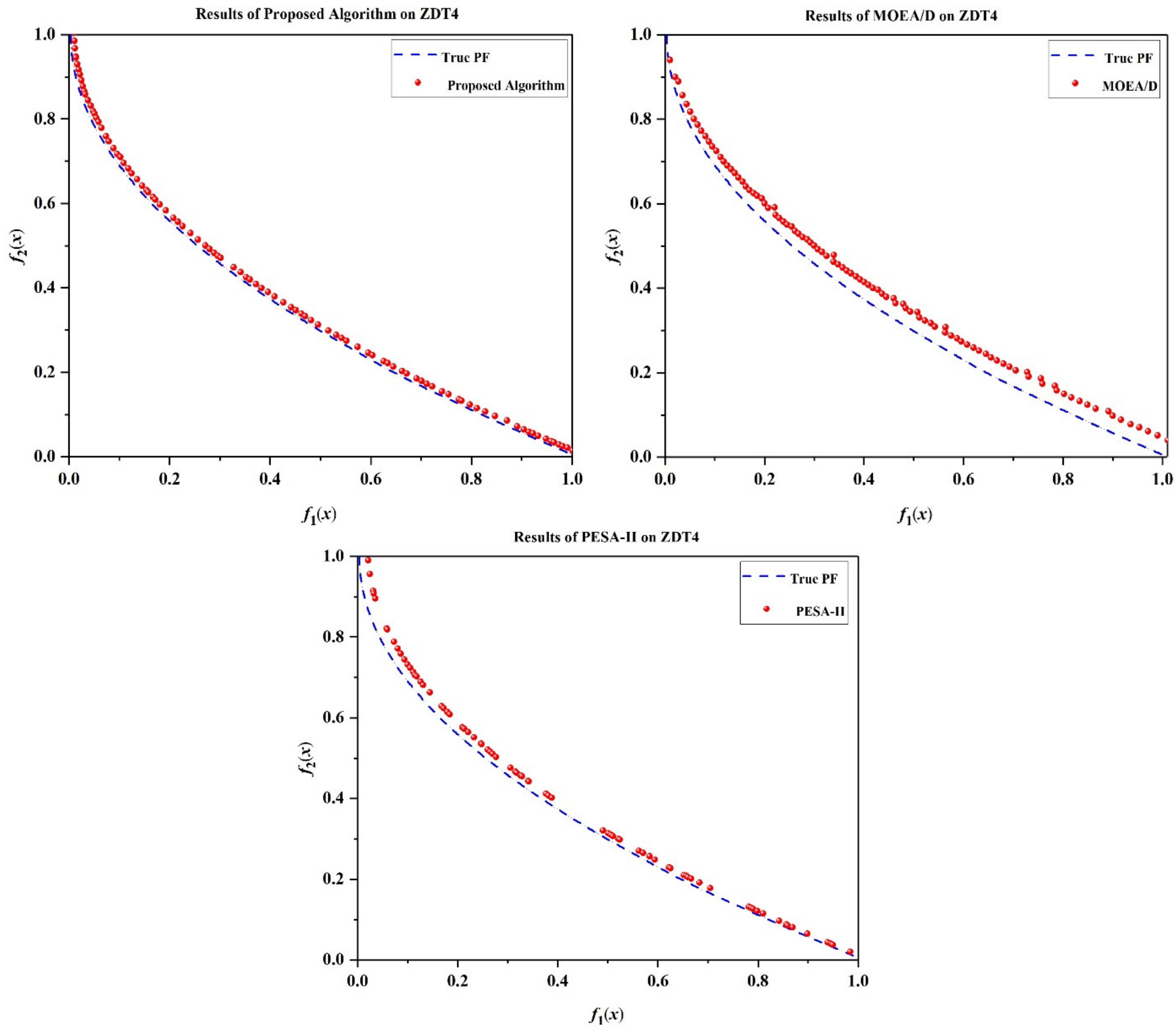


Figure 10: Pareto frontiers of the $ZDT4$ benchmark function produced by $PESA-II$, $MOEA/D$, and the Proposed Algorithm.

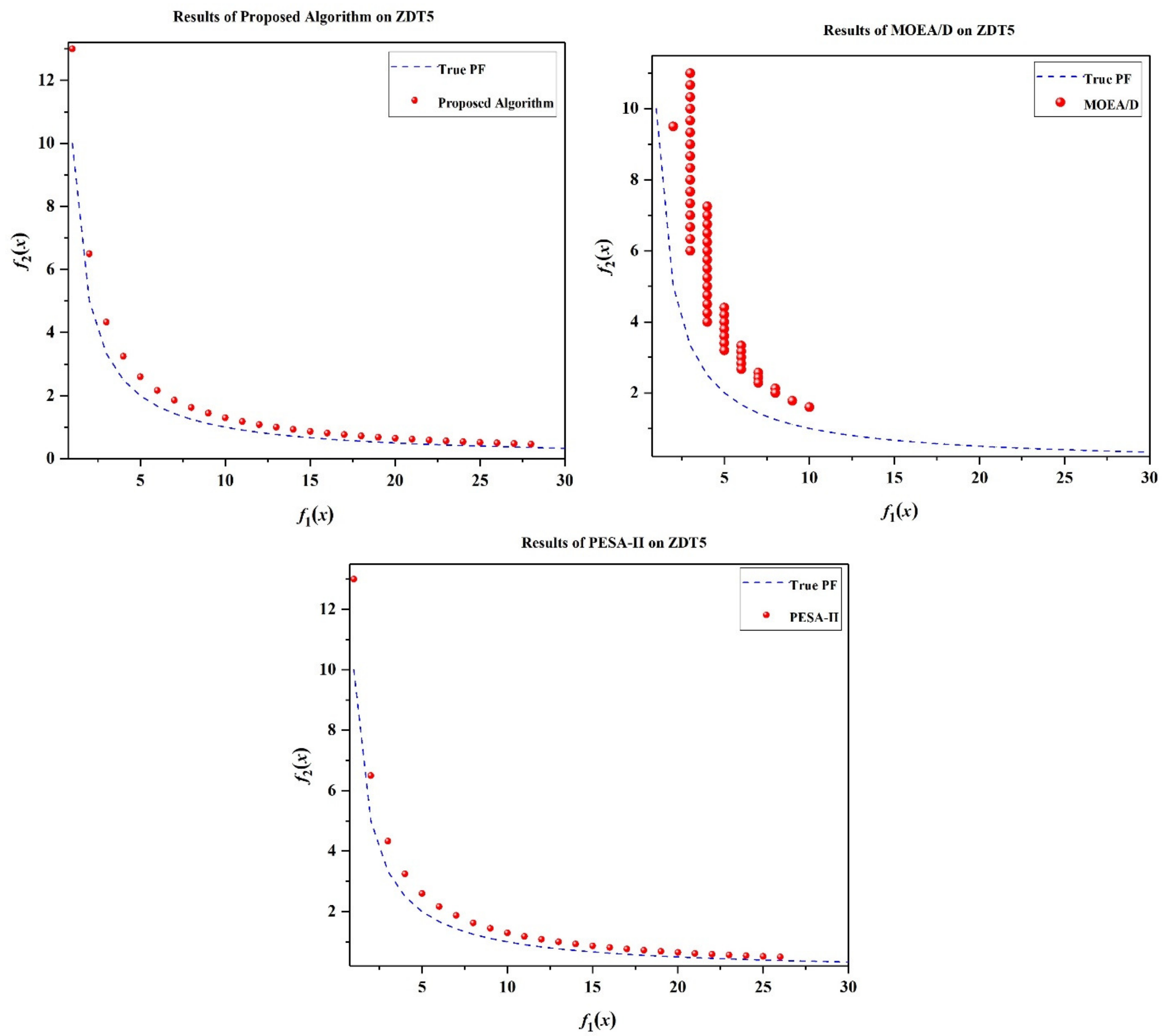


Figure 11: Pareto frontiers of the $ZDT5$ benchmark function produced by $PESA-II$, $MOEA/D$, and the Proposed Algorithm.

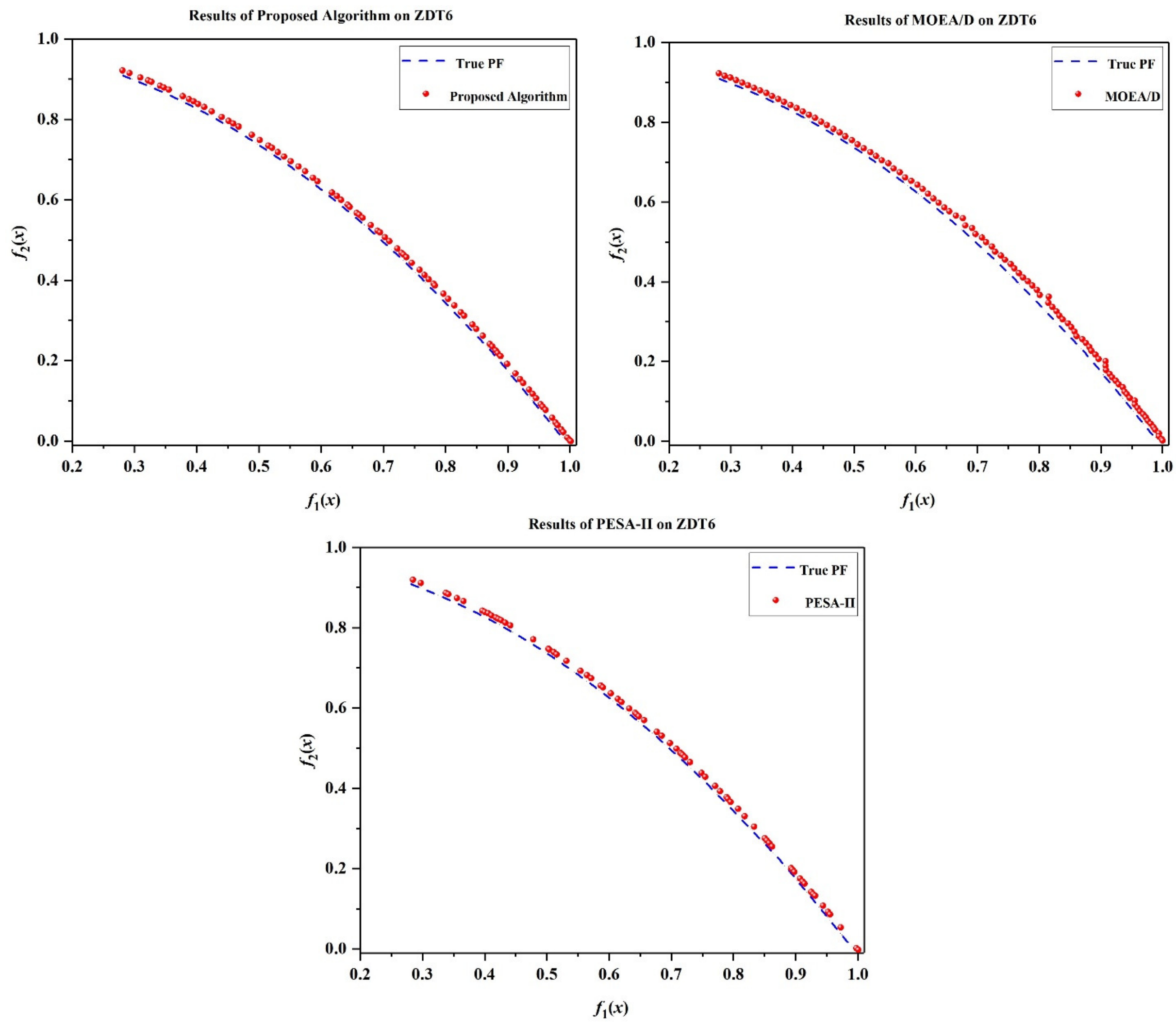


Figure 12: Pareto frontiers of the $ZDT6$ benchmark function produced by $PESA-II$, $MOEA/D$, and the Proposed Algorithm

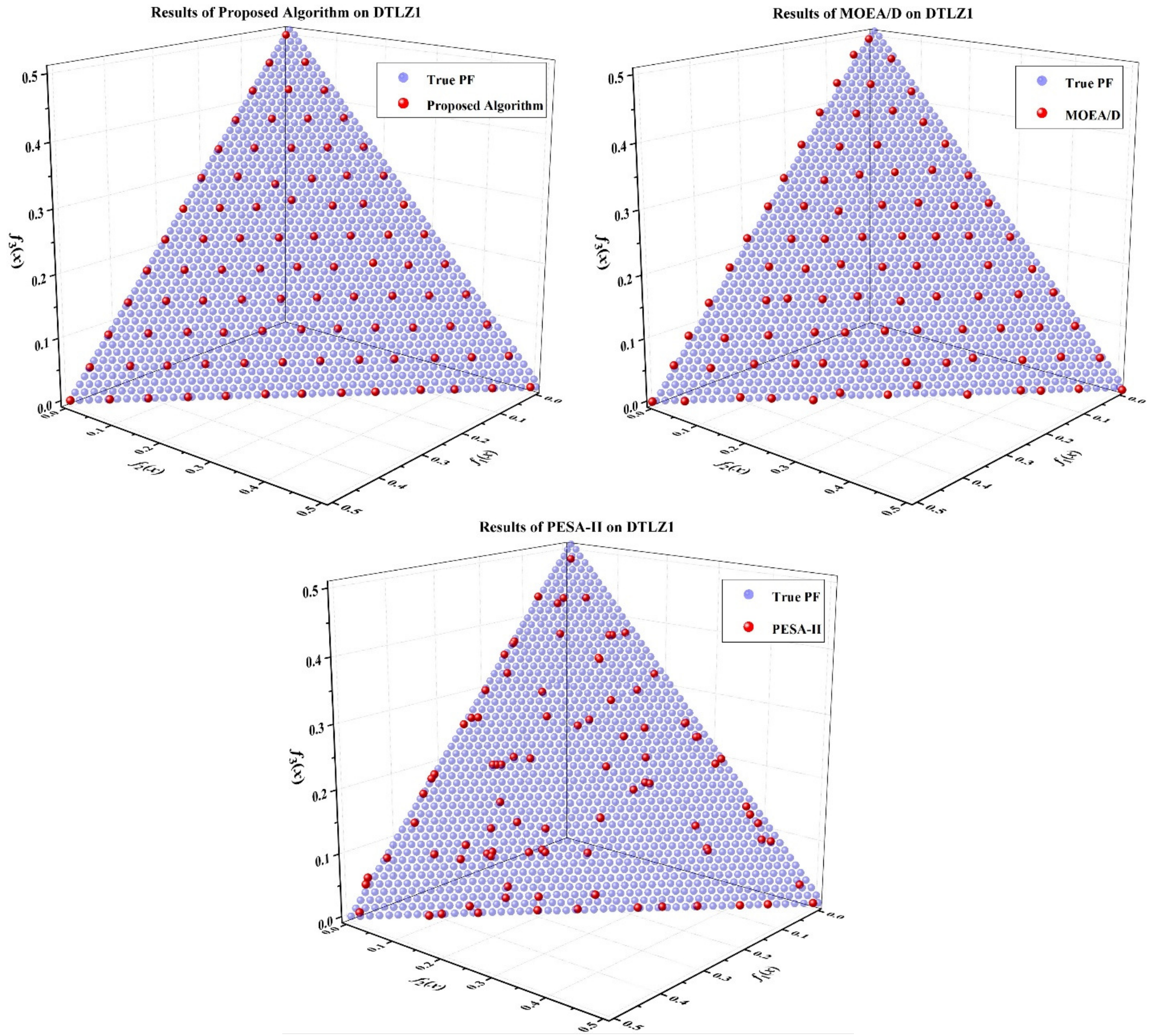


Figure 13: Pareto frontiers of the $DTLZ1$ benchmark function produced by $PESA-II$, $MOEA/D$, and the Proposed Algorithm

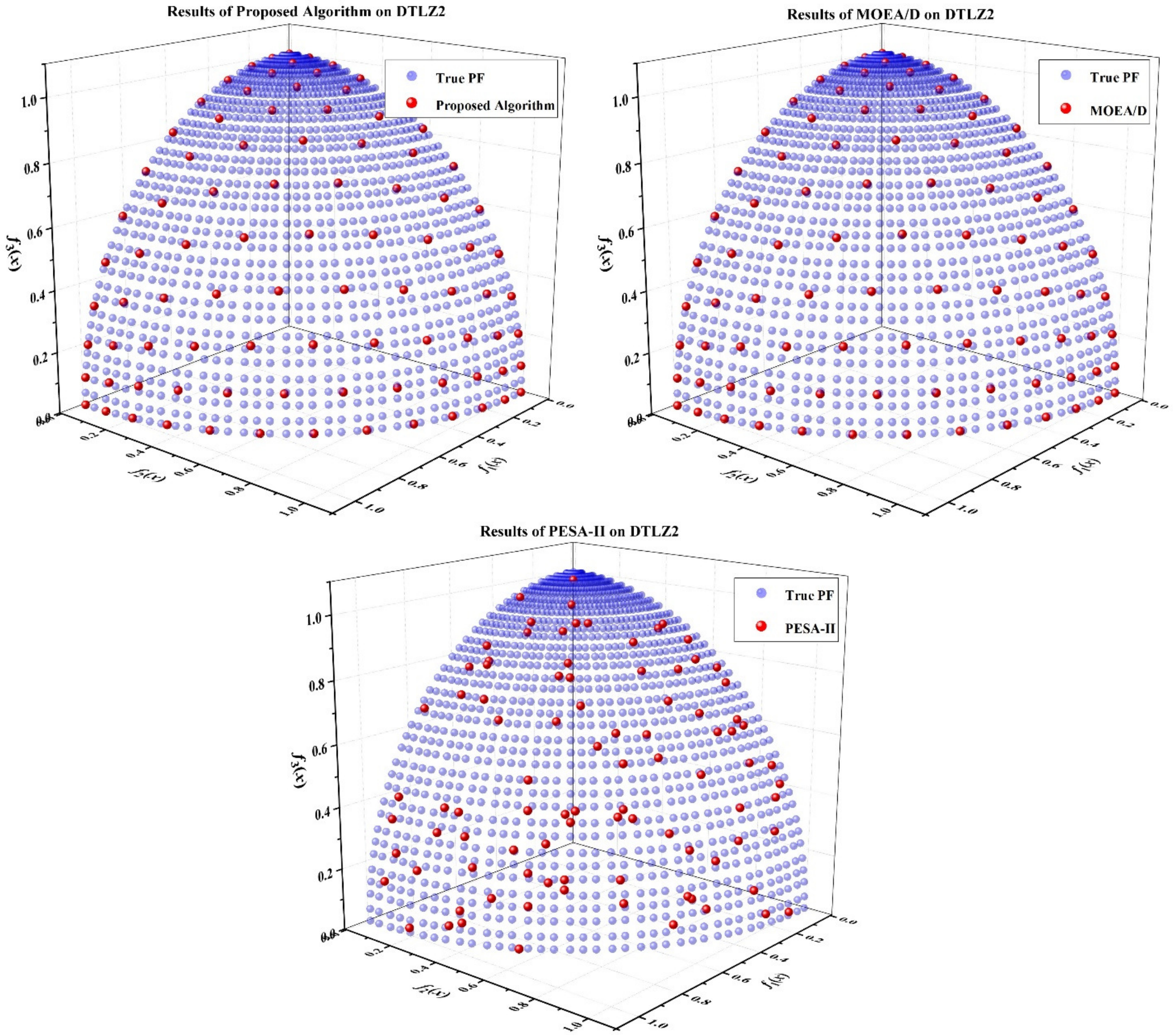


Figure 14: Pareto frontiers of the $DTLZ2$ benchmark function produced by $PESA-II$, $MOEA/D$, and the Proposed Algorithm

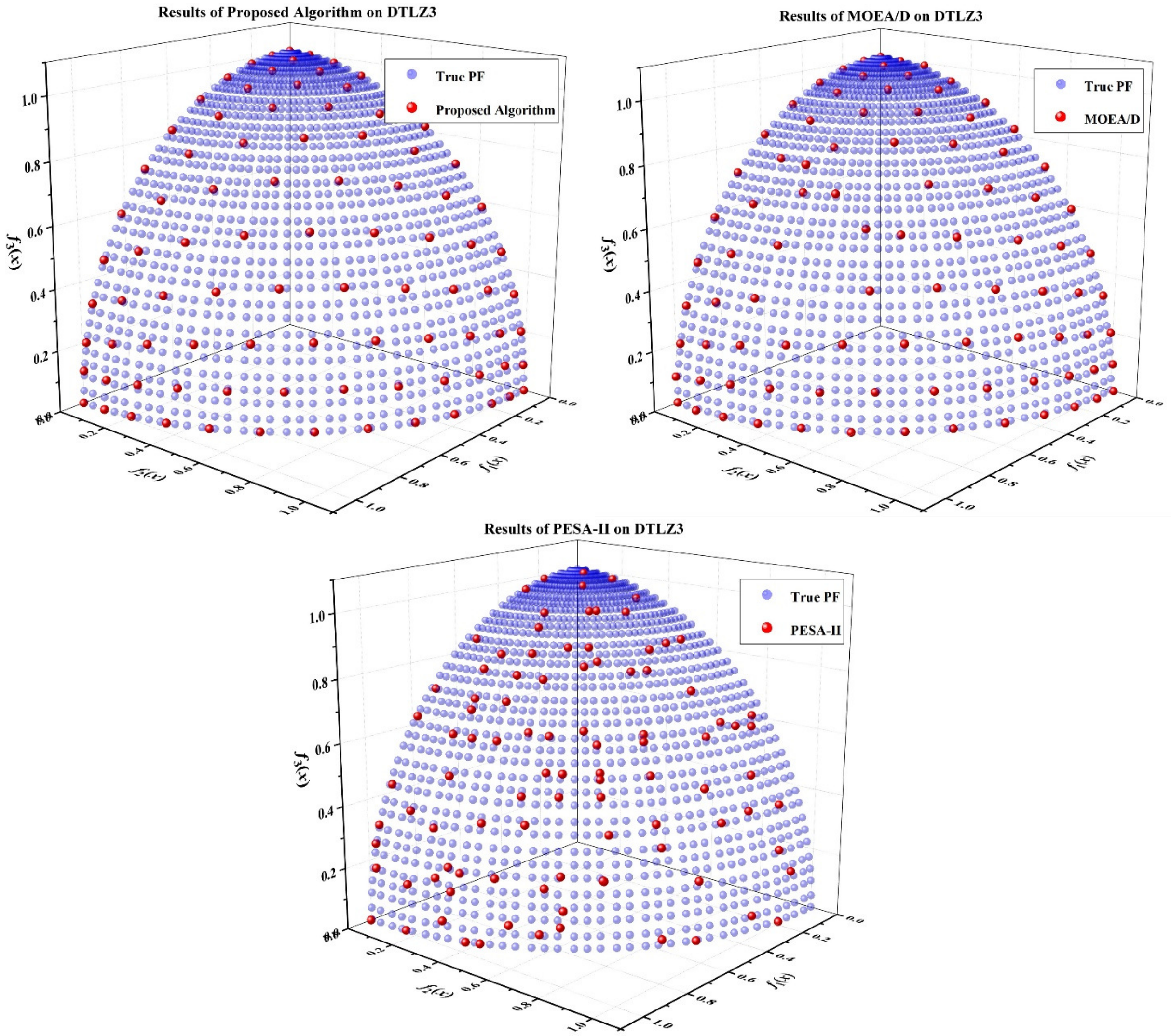


Figure 15: Pareto frontiers of the $DTLZ3$ benchmark function produced by $PESA-II$, $MOEA/D$, and the Proposed Algorithm

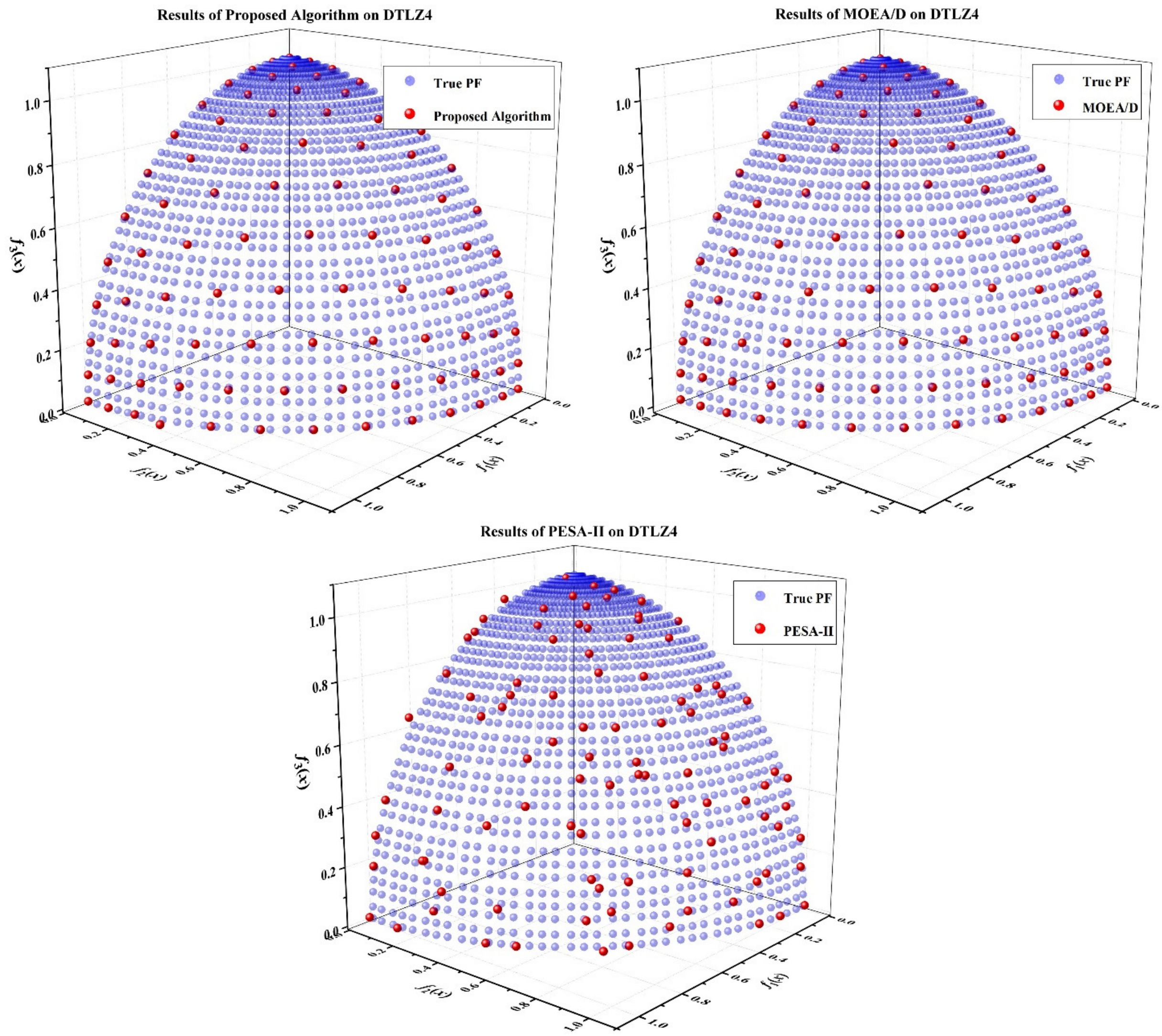


Figure 16: Pareto frontiers of the $DTLZ4$ benchmark function produced by $PESA-II$, $MOEA/D$, and the Proposed Algorithm

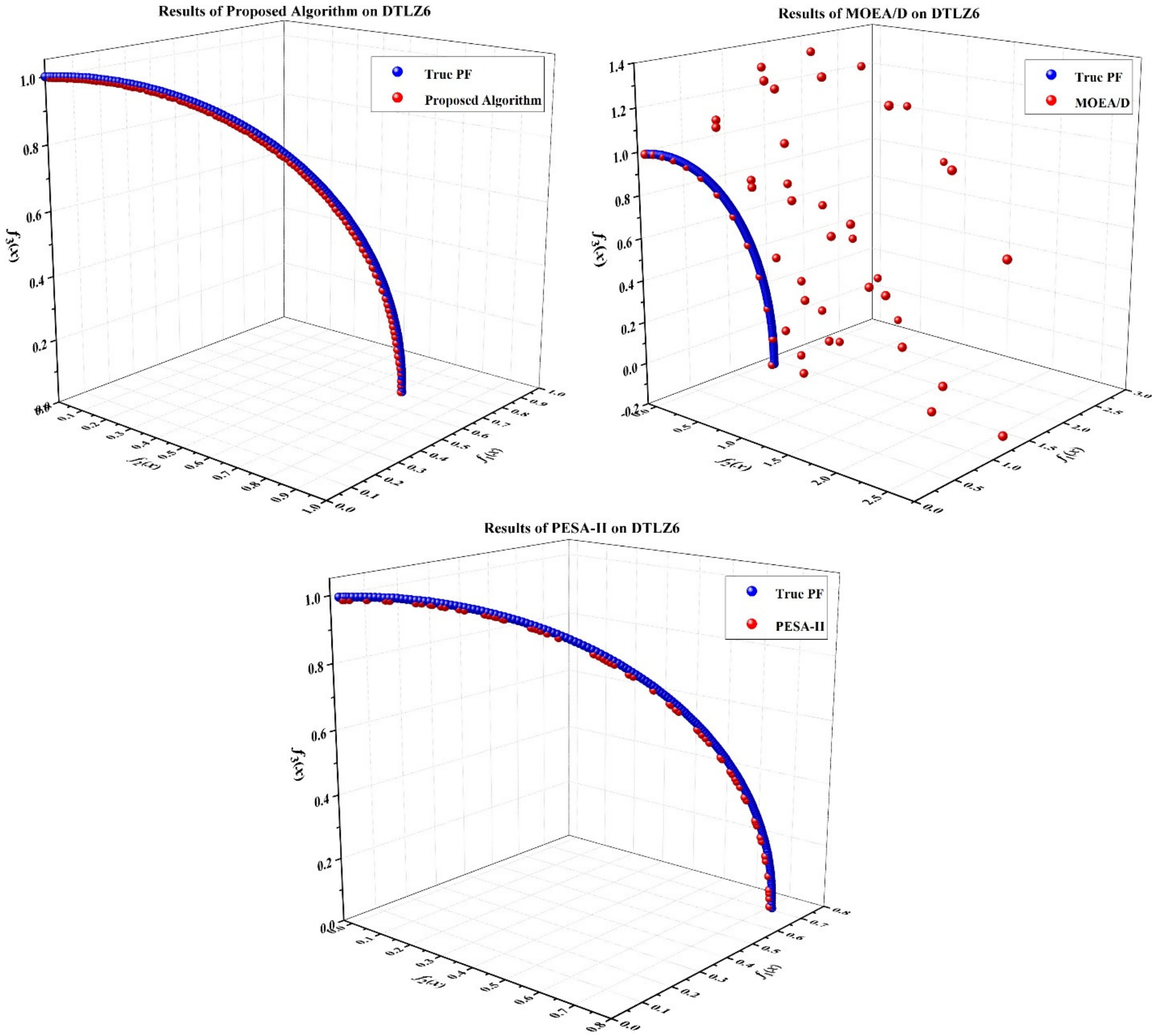


Figure 17: Pareto frontiers of the $DTLZ6$ benchmark function produced by $PESA-II$, $MOEA/D$, and the Proposed Algorithm

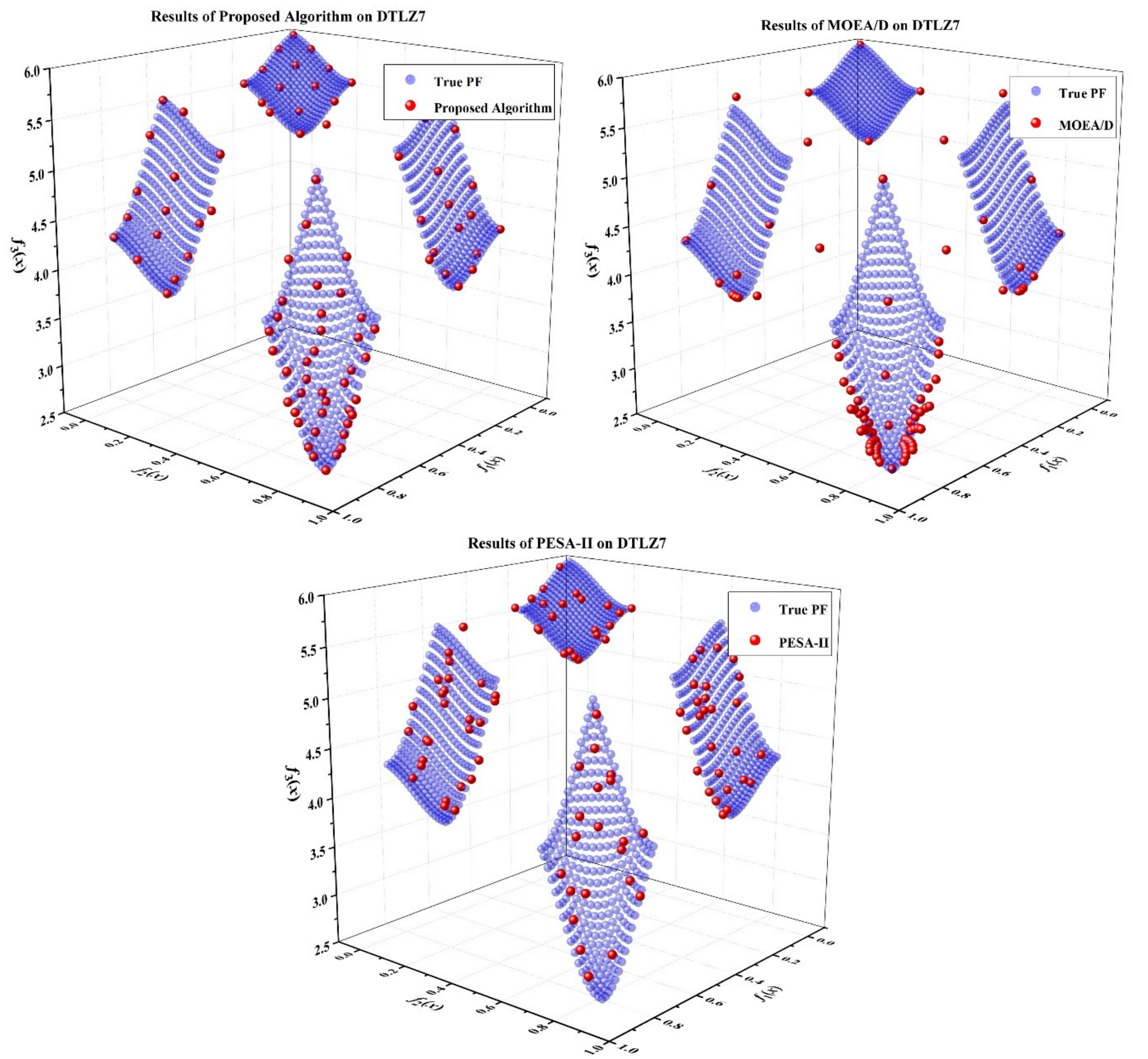


Figure 18: Pareto frontiers of the $DTLZ7$ benchmark function produced by $PESA-II$, $MOEA/D$, and the Proposed Algorithm

Table 6: Result of spread ($\Delta$) on the $ZDT$ bi-objective and $DTLZ$ tri-objective test function suites

| Benchmark Functions | | NSGA-II | MOEA/D | SPEA2 | PESA-II | GrEA | KnEA | Proposed Algorithm |
|---|---|---|---|---|---|---|---|---|
| ZDT1 | Mean | 3.822E-1 | 6.464E-1 | 1.673E-1 | 8.831E-1 | 8.671E-1 | 7.643E-1 | 1.280E-1 |
| | SD | 5.084E-2 | 2.622E-1 | 1.883E-2 | 1.051E-1 | 1.514E-1 | 4.872E-2 | 1.733E-2 |
| ZDT2 | Mean | 4.597E-1 | 6.787E-1 | 2.912E-1 | 8.436E-1 | 8.169E-1 | 8.221E-1 | 2.301E-1 |
| | SD | 1.181E-1 | 3.802E-1 | 1.343E-1 | 1.023E-1 | 8.754E-2 | 6.562E-2 | 1.631E-1 |
| ZDT3 | Mean | 5.104E-1 | 7.609E-1 | 2.579E-1 | 9.416E-1 | 1.012E+0 | 8.163E-1 | 4.289E-1 |
| | SD | 1.423E-1 | 1.374E-1 | 9.971E-2 | 5.654E-2 | 8.242E-2 | 5.781E-2 | 1.093E-1 |
| ZDT4 | Mean | 3.784E-1 | 9.042E-1 | 1.767E-1 | 8.813E-1 | 1.222E+0 | 9.843E-1 | 1.176E-1 |
| | SD | 4.991E-2 | 2.524E-1 | 2.903E-2 | 1.461E-1 | 4.992E-1 | 3.843E-1 | 2.192E-2 |
| ZDT5 | Mean | 1.017E+0 | 1.002E+0 | 1.008E+0 | 1.148E+0 | 1.141E+0 | 1.201E+0 | 1.246E+0 |
| | SD | 5.823E-2 | 9.424E-3 | 4.262E-2 | 1.185E-1 | 5.082E-2 | 5.923E-2 | 2.254E-1 |
| ZDT6 | Mean | 3.498E-1 | 2.459E-1 | 1.759E-1 | 1.073E+0 | 8.626E-1 | 7.469E-1 | 1.069E-1 |
| | SD | 4.331E-2 | 7.003E-2 | 3.474E-2 | 4.093E-1 | 3.143E-1 | 6.192E-2 | 3.114E-2 |
| DTLZ1 | Mean | 5.003E-1 | 5.848E-2 | 1.332E-1 | 9.142E-1 | 1.307E+0 | 8.671E-1 | 1.172E-2 |
| | SD | 7.234E-2 | 2.234E-2 | 2.292E-1 | 5.261E-1 | 5.074E-1 | 5.253E-1 | 1.704E-2 |
| DTLZ2 | Mean | 5.001E-1 | 1.721E-1 | 9.044E-2 | 4.548E-1 | 1.110E+0 | 4.234E-1 | 1.215E-1 |
| | SD | 3.863E-2 | 1.434E-3 | 8.291E-3 | 4.184E-2 | 6.453E-2 | 4.702E-2 | 2.083E-2 |
| DTLZ3 | Mean | 7.986E-1 | 7.604E-1 | 7.760E-1 | 8.316E-1 | 8.248E-1 | 7.849E-1 | 6.594E-1 |
| | SD | 3.363E-1 | 4.214E-1 | 4.991E-1 | 3.992E-1 | 2.414E-1 | 1.023E-1 | 1.401E-1 |
| DTLZ4 | Mean | 4.947E-1 | 6.013E-1 | 1.291E-1 | 7.628E-1 | 1.089E+0 | 4.304E-1 | 1.271E-1 |
| | SD | 3.182E-2 | 4.174E-1 | 1.343E-1 | 3.601E-2 | 6.733E-2 | 5.152E-2 | 1.314E-1 |
| DTLZ6 | Mean | 6.653E-1 | 1.927E+0 | 1.304E-1 | 1.203E-1 | 1.987E+0 | 4.162E-1 | 1.103E-1 |
| | SD | 5.693E-2 | 1.774E-1 | 9.162E-3 | 1.014E-1 | 2.671E-2 | 7.482E-2 | 1.073E-2 |
| DTLZ7 | Mean | 4.874E-1 | 1.060E+0 | 1.167E-1 | 5.607E-1 | 1.018E+0 | 4.583E-1 | 5.488E-1 |
| | SD | 3.213E-2 | 8.342E-2 | 8.224E-3 | 4.263E-2 | 6.531E-2 | 3.634E-2 | 3.982E-2 |

The mean and standard deviation of the $GD$ values on the $ZDT$ and $DTLZ$ test function suites for the seven compared $MOEAs$ are presented in Table 3. In terms of this metric, as can be clearly observed from this Table, the proposed algorithm with the lowest $GD$ outperforms in the majority of the $ZDT$ bi-objective test functions. For example, among the six $ZDT$ test problems, the proposed algorithm achieved the smallest $GD$ values on four bi-objective test problems ($ZDT1$, $ZDT4$, $ZDT5$, and $ZDT6$). However, the $GrEA$ and $NSGA-II$ algorithms demonstrated better results than other algorithms for $ZDT2$ and $ZDT3$, respectively. In addition, for the $DTLZ$ tri-objective test functions, the proposed algorithm represents the best results for $DTLZ1$, $DTLZ2$, $DTLZ4$, and $DTLZ6$. Moreover, it gives satisfactory results even in cases where this algorithm does not perform the best. For example, it is ranked second after the $KnEA$ algorithm for $DTLZ3$, and third after the $GrEA$ and $KnEA$ algorithms for $DTLZ7$. Based on the above-discussed results about the convergence to the true Pareto front set ($GD$ metric), it can be concluded that the proposed algorithm performs better than the six other algorithms in the majority of the test functions.
As done for the $GD$ metric, the results of the $IGD$ found by the considered algorithms are summarized in Table 4. The $IGD$ value represents the performance of the algorithm in terms of both convergence and diversity of the obtained non-dominated solutions. As can be seen from Table 4, the proposed algorithm performs more effectively than 7 out of 12 test functions. For the $IGD$ metric, while it outperforms the other algorithms for optimizing the $ZDT1$, $ZDT3$, $ZDT4$, and $ZDT5$ test functions, the $SPEA2$ algorithm presents better results for the $ZDT2$ and $ZDT6$. As seen in Table 4, for the $DTLZ$ test problems, the proposed algorithm achieved the smallest $IGD$ values on three tri-objective test problems ($DTLZ2$, $DTLZ4$, and $DTLZ7$). While the $MOEA/D$, $PESA-II$, and $SPEA2$ algorithms achieved better results for $DTLZ1$, $DTLZ3$, and $DTLZ4$, respectively. The obtained results indicate that the proposed algorithm, despite not being able

to outperform all other algorithms in all of the test functions, has acceptable performance in terms of the $IGD$ metric.
Table 5 reports the obtained results for the S metric. The proposed algorithm achieved the best results in terms of this metric in four $ZDT$ test functions. As can be seen from Table 5, the proposed algorithm outperforms the other algorithms in optimizing the $ZDT1$, $ZDT2$, $ZDT4$, and $ZDT6$ test functions, whereas the $SPEA2$ algorithm has the best result for the $ZDT3$ and $ZDT5$ test functions. In addition, in terms of the S performance metric, the proposed algorithm is the superior model for four tri-objective test functions ($DTLZ1$, $DTLZ2$, $DTLZ4$, and $DTLZ7$). While the $KnEA$ and $SPEA2$ algorithms achieve the best results for $DTLZ3$ and $DTLZ6$, respectively. Compared to all these algorithms, the obtained results for the $S$ metric show that the proposed algorithm is able to evolve a diverse solution set, resulting in the lowest value of the S metric for most of the test functions.
The obtained results for the $\Delta$ metric, which evaluates the algorithm in terms of diversity and spread, are demonstrated in Table 6. In an overall analysis of this table, it can be seen that the proposed algorithm is superior for the $ZDT1$, $ZDT2$, $ZDT4$, and $ZDT6$ functions. While the $SPEA2$ and $MOEA/D$ algorithms produce the best results for $ZDT3$ and $ZDT5$, respectively. Furthermore, with the exception of the $DTLZ7$ function, the proposed algorithm outperforms the other algorithms in terms of the $\Delta$ metric. In the case of the $DTLZ7$ function, the $SPEA2$ algorithm achieved the best results. This analysis indicates that the proposed algorithm provides the sufficient variety for the Pareto optimal sets.
In an overview of the analysis of the results in Tables 3, 4, 5, and 6, it can be confirmed that the proposed algorithm is better than the six other studied algorithms, especially in $GD$ and $\Delta$ metrics. In order to show this superiority, graphical comparisons between the true Pareto fronts and the approximate Pareto fronts obtained by the proposed, $PESA-II$ and $MOEA/D$ algorithms in solving each test function are shown in Figures $7 \sim 18$. Figures 7 and 8 visually display that for the $ZDT1$ and $ZDT2$ test functions, the proposed algorithm is able to generate solutions that are close to the true Pareto front and well spread along it. The $MOEA/D$ algorithm presents good results close to the true Pareto front, but this algorithm has a poor distribution at the right and left ends of the curve. However, the $PESA-II$ algorithm could not cover the optimal Pareto front well and provide a uniform distribution. For the $ZDT3$ test function, the proposed algorithm discovers accurate results, but the $MOEA/D$ and $PESA-II$ algorithms are unable to produce a set of solutions that have good convergence and diversity. For the $ZDT4$ test function, the proposed algorithm is much more successful than the $MOEA/D$ and $PESA-II$, while the $MOEA/D$ and $PESA-II$ algorithms are similar in that they generate solutions close to the true Pareto front. Nevertheless, it is clear that the $PESA-II$ has a poor distribution. In the $ZDT5$ test function, the general behavior of the proposed algorithm is similar to that of the $PESA-II$; however, the performance of the $MOEA/D$ algorithm is not acceptable. The results for the $ZDT6$ test function show that all algorithms produce solutions close to the true Pareto front and well distributed along it, while the proposed algorithm presents slightly better solutions. Similar

results are obtained for the tri-objective test functions, i.e. $DTLZ1$, $DTLZ2$, $DTLZ3$, and $DTLZ4$. As seen in Figures 13 $\sim$ 16, the proposed algorithm presents quite good results and $MOEA/D$ is much more successful than the $PESA-II$ algorithm. For the $DTLZ6$ test function, the proposed algorithm is able to achieve better results than the other algorithms, considering both convergence and diversity, while $MOEA/D$ produces the worst results in general. In the optimization of the $DTLZ7$ test function, the proposed algorithm has resulted in the best convergence and diversity compared to the other algorithms.

# 6 Conclusion

Many real-world problems have more than one objective function to be optimized simultaneously. In the present paper, an improved imperialistic competitive algorithm to solve multi-objective optimization problems has been presented, through modifying the convergence and diversity of solutions. In order to increase the diversity of solutions, the proposed method has applied the Sigma method in empire formation and the $WSum$ method in the competition between empires. To evaluate the performance of the proposed algorithm, 12 benchmark functions, including six bi-objective $ZDT$ functions and six tri-objective $DTLZ$ functions, have been analyzed by considering four performance metrics. In comparison with six multi-objective algorithms: $NSGA-II$, $MOEA/D$, $SPEA2$, $PESA-II$, $GrEA$, and $KnEA$, results have demonstrated that the proposed algorithm significantly outperforms the $MOEA/D$ and $PESA-II$ algorithms. Because the Pareto front produced by the proposed algorithm was more regular in most cases and lead to better convergence than the $MOAE/D$ and $PESA-II$ algorithms. The results have also shown that this algorithm is comparable to or is even better than other algorithms like the $NSGA-II$, $SPEA2$, $GrEA$, and $KnEA$. Moreover, it has achieved the desirable solutions in most of the benchmark functions. The results have indicated that, in general, the approach is superior to other algorithms not only in approximating the Pareto optimal front but also in terms of diversity and distribution of solutions in most of the test problems.

# 7 Acknowledgments

This research did not receive any specific grant from funding agencies in the public, commercial, or not-for-profit sectors.

# References

[1] Aliniya Z., Keyvanpour M. R., CB-ICA: A crossover-based imperialist competitive algorithm for large-scale problems and engineering design optimization, Neural. Comput. Appl. 31 (2019) 7549–7570.

[2] Arya Y., Impact of ultra-capacitor on automatic generation control of electric energy systems using an optimal FFOID controller, Int. J. Energy Res. 43 (2019) 8765–8778.

[3] Ashry G., Onglobally convergent multi-objective optimization, Appl. Math. Comput. 183 (2006) 209–216.

[4] Atashpaz-Gargari E., Lucas C., Imperialist competitive algorithm: An algorithm for optimization inspired by imperialistic competition. 2007 IEEE Congress on Evolutionary Computation, Singapore, (2007)

[5] Branke J., Mostaghim S., About Selecting the Personal Best in Multi-Objective Particle Swarm Optimization, In Lecture Notes in Computer Science, Berlin, Heidelberg, Springer, 523–532 (2006)

[6] Coello Coello C., Lechunga M., MOPSO:aproposal for multiple objective particle swarm optimization, In IEEE World Congress on Computational Intelligence, IEEE, Hawaii (2000)

[7] Coello Coello C., Cortes N., Solving Multiobjective Optimization Problems Using an Artificial Immune System, Genet. Program. Evolvable Mach. 6 (2005)163–190.

[8] Corne D., Jerram N., Knowles J., PESA-II: Region-based selection in evolutionary multiobjective optimization, In Proceedings of the Genetic and Evolutionary Computation Conference (GECCO), San Francisco, CA, USA, (2001)

[9] Deb K., Agrawal S., Pratap A., Meyarivan T., A Fast Elitist Non-dominated Sorting Genetic Algorithm for Multi-objective Optimisation: NSGA-II, In Proceedings of the Parallel Problem Solving from Nature VI Conference, Paris, France (2000)

[10] Deb K., Pratap A., Agarwal S., A fast and elitist multiobjective genetic algorithm: NSGA-II. IEEE Trans. Evol. 6 (2002) 182 – 197.

[11] Deb K., Thiele L., Laumanns M., Zitzler E., Scalable Test Problems for Evolutionary Multiobjective Optimization. in: A. Abraham, L. Jain, R. Goldberg (Eds), Evolutionary Multiobjective Optimization, Springer, London, 2005, pp. 105–145.

[12] Elsisi M., Design of neural network predictive controller based on imperialist competitive algorithm for automatic voltage regulator, Neural. Comput. Appl. 31(2019) 5017–5027.

[13] Enayatifar R., Yousefi M., Abdullah A. H., Darus A. N., MOICA: Anovel multiobjective approach based onimperialist competitive algorithm, Appl. Math. Comput. 219 (2013)8829–8841.

[14] Fakhrzad M., Goodarzian F., A Fuzzy multi-objective programming approach to develop a green closed-loop supply chain network design problem under uncertainty: Modifications of imperialist competitive algorithm, RAIRO Oper. Res. 53 (2019) 963–990.

[15] Gharib M., Mohammad S., A dynamic dispatching problem to allocate relief vehicles after a disaster, Eng. Optim. 53 (2020) 1999–2016.

[16] Ghasemi M., Ghavidel S., Ghanbarian M., Gitizadeh M., Multi-objective optimal electric power planning in the power system using Gaussian bare-bones imperialist competitive algorithm, Inf. Sci. 294 (2015) 286-304.

[17] Hajiaghaei-Keshteli M.,Fard A ., Sustainable closed-loop supply chain network design with discount supposition, Neural. Comput. Appl. 31 (2019) 5343–5377.

[18] Hosseini S., Khaled A. A., A survey on the Imperialist Competitive Algorithm metaheuristic Implementation in engineering domain and directions for future research, Appl. Soft Comput. 24 (2014) 1078-1094.

[19] Hosseinzadeh A. Z., Razzaghi S. R. S., Amiri G. G., An iterated IRS technique for cross-sectional damage modelling and identification in beams using limited sensors measurement, Inverse Probl. Sci. Eng. 27 (2019) 1145–1169.

[20] Iyer V. H., Mahesh S., Malpani R., Sapre M., Adaptive Range Genetic Algorithm: A hybrid optimization approach and its application in the design and economic optimization of Shell-and-Tube Heat Exchanger, Eng. Appl. Artif. Intell. 85 (2019) 444–461.

[21] Karimi B., Hassanlu M., Niknamfar A., An integrated production-distribution planning with a routing problem and transportation cost discount in a supply chain, Assem. Autom. 39 (2019)783-802.

[22] Kaveh A., Talatahari S., Optimum design of skeletal structures using imperialist competitive algorithm, Comput. Struct. 88 (2010) 1220–1229.

[Khanali[23] Khanali M., Akram A., Behzadi J., Mostashari-Rad F., Saber Z., Chau K.-w., Nabavi-Pelesaraei A., Multi-objective optimization of energy use and environmental emissions for walnut production using imperialist competitive algorithm, Appl. Energy 284 (2021) 116342.

[24] Lei D., Li M., Wang L., A two-phase meta-heuristic for multi-objective flexible job shop scheduling problem with total energy, IEEE Trans. Cybern. 49 (2018) 1097–1109.

[25] Li X., Wong H.-S. , Logic optimality for multi-objective optimization, Appl. Math. Comput. 215 (2009) 3045–3056.

[26] Lucas C., Nasiri-Gheidari Z., Tootoonchian F., Application of an imperialist competitive algorithm to the design of a linear induction motor, Energy Conv. Manag. 51 (2010) 1407–1411.

[27] Luo J., Zhou J., Jiang X., Lv H., A Modification of the Imperialist Competitive Algorithm with Hybrid Methods for Multi-Objective Optimization Problems, Sym. 14 (2022) 173.

[28] Mahmoodabadi M. J., Bagheri A., Nariman-zadeh N., Jamali A., A new optimization algorithm based on a combination of particle swarm optimization, convergence and divergence operators for single-objective and multi-objective problems, Eng. Optim. 44 (2012)1167-1186.

[29] Marandi F., Fatemi Ghomi S., Integrated multi-factory production and distribution scheduling applying vehicle routing approach, Int. J. Prod. Res. 57 (2019) 722-748.

[30] Mostaghim S., Teich J., Strategies for finding good local guides in multi-objective particle swarm optimization (MOPSO). Proceedings of the 2003 IEEE Swarm Intelligence Symposium, SIS'03 (Cat. No.03EX706), 26-33 (2003)

[31] Nazari-Shirkouhi S., Eivazy H., Ghodsi R., Rezaie K., Atashpaz-Gargari E., Solving the integrated product mix-outsourcing problem using the Imperialist Competitive Algorithm, Expert Syst. Appl. 37 (2010) 7615–7626.

[32] Nejlaoui M., Houidi A., Affi Z., Romdhane L., A hybrid multi-objective imperialist competitive algorithm and Monte Carlo method for robust safety design of a rail vehicle, C. R. - Mec. 345 (2017) 712-723.

[33] Piroozfard H., Wong K., Tiwari M., Reduction of carbon emission and total late work criterion in job shop scheduling by applying a multi-objective imperialist competitive algorithm, Int. J. Comput. Intell. 11 (2018) 805 – 829.

[34] Schott J. R., Fault tolerant design using single and multi-criteria genetic algorithms, Master's thesis, Cambridge, Massachusetts: Department of Aeronautics and Astronautics, Massachusetts Institute of Technology (1995)

[35] Sharifi M. A., Mojallali H., Multi-Objective Modified Imperialist Competitive Algorithm for Brushless DC Motor Optimization, IETE J. Res. 65 (2019) 96-103.

[36] Shengxiang Y., Miqing L., Xiaohui L., Jinhua Z., A Grid-Based Evolutionary Algorithm for Many-Objective Optimization, IEEE Trans. Evol. 17 (2013) 721 – 736.

[37] VanVeldhuizen D., Lamont G., Evolutionary computation and convergence to a Pareto front, In Late Breaking Papers at the Genetic Programming 1998 Conference, University of Wisconsin, Madison, Wisconsin, USA, Stanford University Bookstore (1998)

[38] Wang S., Liu G., Gao S., A Hybrid Discrete Imperialist Competition Algorithm for Fuzzy Job-Shop Scheduling Problems, IEEE Access. 4 (2017) 9320–9331.

[39] Xingyi Z., Zhang Y., Yaochu T. J., A Knee Point-Driven Evolutionary Algorithm for Many-Objective Optimization, IEEE Trans. Evol. 19 (2015) 761 – 776.

[40] Yousefi M., Darus A., Mohammadi H., An imperialist competitive algorithm for optimal design of plate-fin heat exchangers, Int. J. Heat Mass Transf. 55 (2012) 3178-3185.

[41] Zhang Q., Li H., MOEA/D: A Multiobjective Evolutionary Algorithm Based on Decomposition, IEEE Trans. Evol. 11 (2007) 712–731.

[42] Zhang H., Balancing Problem of Stochastic Large-Scale U-Type Assembly Lines Using a Modified Evolutionary Algorithm, IEEE Access. 6 (2018)78414–78424.

[43] Zitzler E., Deb K., Thiele L., Comparison of Multi-objective Evolutionary Algorithms: Empirical Results, Evol. Comput. 8 (2000) 173–195.

[44] Zitzler E. , Laumanns M., Thiele L., SPEA2: Improving the strength pareto evolutionary algorithm, ETH Zurich, Computer Engineering and Networks Laboratory, Switzerland, TIK Report 103 (2001)