Expert Systems With Applications 47 (2016) 106–119

Contents lists available at ScienceDirect

Expert Systems With Applications

journal homepage: www.elsevier.com/locate/eswa

Multi-objective grey wolf optimizer: A novel algorithm for multi-criterion optimization

Seyedali Mirjalili [a,b,*], Shahrzad Saremi [a,b], Seyed Mohammad Mirjalili [c], Leandro dos S. Coelho [d]

[a] School of Information and Communication Technology, Griffith University, Nathan Campus, Brisbane, QLD 4111, Australia
[b] Griffith College, Mt Gravatt, Brisbane, QLD 4122, Australia
[c] Independent Researcher, Tehran, Iran
[d] Industrial and Systems Engineering Graduate Program (PPGEPS), Pontifical Catholic University of Parana (PUCPR), Curitiba, Parana, Brazil

ARTICLE INFO

Keywords:
Multi-objective optimization
Evolutionary algorithm
Multi-criterion optimization
Heuristic algorithm
Meta-heuristic
Engineering optimization
Grey wolf optimizer

ABSTRACT

Due to the novelty of the Grey Wolf Optimizer (GWO), there is no study in the literature to design a multi-objective version of this algorithm. This paper proposes a Multi-Objective Grey Wolf Optimizer (MOGWO) in order to optimize problems with multiple objectives for the first time. A fixed-sized external archive is integrated to the GWO for saving and retrieving the Pareto optimal solutions. This archive is then employed to define the social hierarchy and simulate the hunting behavior of grey wolves in multi-objective search spaces. The proposed method is tested on 10 multi-objective benchmark problems and compared with two well-known meta-heuristics: Multi-Objective Evolutionary Algorithm Based on Decomposition (MOEA/D) and Multi-Objective Particle Swarm Optimization (MOPSO). The qualitative and quantitative results show that the proposed algorithm is able to provide very competitive results and outperforms other algorithms. Note that the source codes of MOGWO are publicly available at http://www.alimirjalili.com/GWO.html.

1. Introduction

There are different challenges in solving real engineering problems, which needs specific tools to handle them. One of the most important characteristics of real problems, which make them challenging, is multi-objectivity. A problem is called multi-objective if there is more than one objective to be optimized. Needless to say, a multiple objective optimizer should be employed in order to solve such problems. There are two approaches for handling multiple objectives: *a priori* versus *a posteriori* (Branke, Kaußler, & Schmeck, 2001; Marler & Arora, 2004).

The former class of optimizers combines the objectives of a multi-objective problem to a single-objective with a set of weights (provided by decision makers) that defines the importance of each objective and employs a single-objective optimizer to solve it. The unary-objective nature of the combined search spaces allows finding a single solution as the optimum. In contrary, *a posterior* method maintain the multi-objective formulation of multi-objective problems, allowing to explore the behavior of the problems across a range of design parameters and operating conditions compared to *a priori* approach (Deb, 2012). In this case, decision makers will eventually choose one of the obtained solutions based on their needs. There is also another type of handling multiple objectives called progressive method, in which decision makers' preferences about the objectives are considered during optimization (Branke & Deb, 2005).

In contrary to single-objective optimization, there is no single solution when considering multiple objectives as the goal of the optimization process. In this case, a set of solutions, which represents various trade-offs between the objectives, includes optimal solutions of a multi-objective problem (Coello, Lamont, & Van Veldhuisen, 2007). Before 1984, mathematical multi-objective optimization techniques were popular among researchers in different fields of study such as applied mathematics, operation research, and computer science. Since the majority of the conventional approaches (including deterministic methods) suffered from stagnation in local optima, however, such techniques were not applicable as there are not nowadays.

In 1984, a revolutionary idea was proposed by David Schaffer (Coello Coello, 2006). He introduced the concepts of multi-objective optimization using stochastic optimization techniques (including evolutionary and heuristic). Since then, surprisingly, a significant number of researches have been dedicated for developing and

* Corresponding author at: School of Information and Communication Technology, Griffith University, Nathan Campus, Brisbane, QLD 4111, Australia. Tel.: +61 434555738.
E-mail addresses: seyedali.mirjalili@griffithuni.edu.au, ali.mirjalili@gmail.com (S. Mirjalili), shahrzad.saremi@griffithuni.edu.au (S. Saremi), mohammad.smm@gmail.com (S.M. Mirjalili), lscoelho2009@gmail.com (L.d.S. Coelho).

http://dx.doi.org/10.1016/j.eswa.2015.10.039
0957-4174/

evaluating multi-objective evolutionary/heuristic algorithms. The advantages of stochastic optimization techniques such as gradient-free mechanism and local optima avoidance made them readily applicable to the real problems as well. Nowadays, the application of multi-objective optimization techniques can be found in different fields of studies: mechanical engineering (Kipouros et al., 2008), civil engineering (Luh & Chueh, 2004), chemistry (Gaspar-Cunha & Covas, 2004; Rangaiah, 2008), and other fields (Coello & Lamont, 2004).

Early year of multi-objective stochastic optimization saw conversion of different single-objective optimization techniques to multi-objective algorithms. Some of the most well-known stochastic optimization techniques proposed so far are as follows:

- Strength–Pareto Evolutionary Algorithm (SPEA) (Zitzler, 1999; Zitzler & Thiele, 1999).
- Non-dominated Sorting Genetic Algorithm (Srinivas & Deb, 1994)
- Non-dominated Sorting Genetic Algorithm version 2 (NSGA-II) (Deb, Pratap, Agarwal, & Meyarivan, 2002)
- Multi-Objective Particle Swarm Optimization (MOPSO) (Coello, Pulido, & Lechuga, 2004)
- Multi-Objective Evolutionary Algorithm based on Decomposition (MOEA/D) (Zhang & Li, 2007)
- Pareto Archived Evolution Strategy (PAES) (Knowles & Corne, 2000)
- Pareto–frontier Differential Evolution (PDE) (Abbass, Sarker, & Newton, 2001).

The literature shows that these algorithms are able to effectively approximate the true Pareto optimal solutions of multi-objective problems. However, there is a theorem here called No Free Lunch (NFL) (Wolpert & Macready, 1997) that has been logically proved that there is no optimization technique for solving all optimization problems. According to this theorem, the superior performance of an optimizer on a class of problems cannot guarantee the similar performance on another class of problems. This theorem is the foundation of many works in the literature and allows researchers in this field to adapt the current techniques for new classes of problems or propose new optimization algorithms. This is the foundation and motivation of this work as well, in which we propose a novel multi-objective optimization algorithm called Multi-Objective Grey Wolf Optimizer (MOGWO) based on the recently proposed Grey Wolf Optimizer (GWO). The contributions of this research are as follows:

- An archive has been integrated to the GWO algorithm to maintain non-dominated solutions.
- A grid mechanism has been integrated to GWO in order to improve the non-dominated solutions in the archive.
- A leader selection mechanism has been proposed based on alpha, beta, and delta wolves to update and replace the solutions in the archive.
- The multi-objective version of GWO has been proposed utilizing the above three operators.

The rest of the paper is organized as follows. Section 2 presents definitions and preliminaries of optimization in a multi-objective search space. Section 3 briefly reviews the concepts of GWO and then proposes the MOGWO algorithm. The qualitative and qualitative results as well as relevant discussion are presented in Section 4. Eventually, Section 5 concludes the work and outlines some advises for future works.

2. Literature review

This section provides the concepts of multi-objective optimization and current techniques in the field of meta-heuristics.

2.1. Multi-objective optimization

As briefly mentioned in the introduction, multi-objective optimization refers to the optimisation of a problem with more than one objective function. Without loss of generality, it can be formulated as a maximization problem as follows:

$$Maximize: \ F(\vec{x}) = f_1(\vec{x}), \ f_2(\vec{x}), \ \ldots, \ f_o(\vec{x}) \tag{2.1}$$

$$Subject\ to: \ g_i(\vec{x}) \geq 0, \ i = 1, 2, \ldots, m \tag{2.2}$$

$$h_i(\vec{x}) = 0, \ i = 1, 2, \ldots, p \tag{2.3}$$

$$L_i \leq x_i \leq U_i, \ i = 1, 2, \ldots, n \tag{2.4}$$

where n is the number of variables, o is the number of objective functions, m is the number of inequality constraints, p is the number of equality constraints, g_i is the ith inequality constraints, h_i indicates the ith equality constraints, and $[L_i,U_i]$ are the boundaries of ith variable.

In single-objective optimization, solutions can be compared easily due to the unary objective function. For maximization problems, solution X is better than Y if and only if $X > Y$. However, the solutions in a multi-objective space cannot be compared by the relational operators due to multi-criterion comparison metrics. In this case, a solution is better than (dominates) another solution if and only if it shows better or equal objective value on all of the objectives and provides a better value in at least one of the objective functions. The concepts of comparison of two solutions in multi-objective problems were first proposed by Francis Ysidro (Edgeworth, 1881) and then extended by Vilfredo Pareto (Pareto, 1964). Without loss of generality, the mathematical definition of Pareto dominance for a maximization problem is as follows (Coello, 2009):

Definition 1. Pareto Dominance:

Suppose that there are two vectors such as: $\vec{x} = (x_1, x_2, \ldots, x_k)$ and $\vec{y} = (y_1, y_2, \ldots, y_k)$.

Vector x dominates vector y (denote as $x \succ y$) *iff*:

$$\forall i \in \{1, 2, \ldots, k\}, \ [f(x_i) \geq f(y_i)] \wedge [\exists i \in 1, 2, \ldots, k : f(x_i)] \tag{2.5}$$

The definition of Pareto optimality is as follows (Ngatchou, Zarei, & El-Sharkawi, 2005):

Definition 2. Pareto Optimality:

A solution $\vec{x} \in X$ is called Pareto-optimal *iff*:

$$\nexists \vec{y} \in X \mid F(\vec{y}) \succ F(\vec{x}) \tag{2.6}$$

A set including all the non-dominated solutions of a problem is called Pareto optimal set and it is defined as follows:

Definition 3. Pareto optimal set:

The set all Pareto-optimal solutions is called Pareto set as follows:

$$P_s := \{x, y \in X \mid \exists F(y) \succ F(x)\} \tag{2.7}$$

A set containing the corresponding objective values of Pareto optimal solutions in Pareto optimal set is called Pareto optimal front. The definition of the Pareto optimal front is as follows:

Definition 4. Pareto optimal front: a set containing the value of objective functions for Pareto solutions set:

$$P_f := \{F(x) | x \in P_s\} \tag{2.8}$$

2.2. Multi-objective meta-heuristics

The ultimate goal of multi-objective optimization algorithms (*a posteriori* methods) is to find very accurate approximation of the true Pareto optimal solutions with the highest diversity (Zhou et al., 2011). This allows decision makers to have a diverse range of design options. In the past, the solution of multi-objective problems would have been undertaken by *a priori* aggregation of objectives into a single objective. However, this method has two main drawbacks (Das & Dennis, 1998; Kim & De Weck, 2005; Messac & Mattson, 2002): An even distribution of the weights does not necessarily guarantee finding Pareto optimal solutions with an even distribution, and this method is not able to find the non-convex regions of Pareto optimal front because the negative weights are not allowed and sum of all the weights should be constant. In other words, the convex sum of the objectives is usually used in the conventional aggregation methods.

There are some works in the literature that tried to improve this method. For example Parsopoulos and Vrahatis used two dynamic weighted aggregations to change the weights gradually over time or abruptly (Parsopoulos & Vrahatis, 2002). However, these methods did not completely solve the main problems of the aggregation method. In addition, these methods need to be run many times to approximate the whole Pareto optimal solutions because there is only one best obtained solution in each iteration. According to Deb (Deb, 2012), the multi-objective optimization process utilizing meta-heuristics deals with overcoming many difficulties such as infeasible areas, local fronts, diversity of solutions, and isolation of optimum. *A priori* methods should deal with all these difficulties in each run. However, maintaining the multi-objective formulation of problems brings some advantages. First, the information about the search space is exchanged between the search agents, which results in quick movements towards the true Pareto optimal front.

Second, the multi-objective approaches assist to approximate the whole true Pareto optimal front in a single run. Finally, maintaining the multi-objective formulation of a problem allows the exploration of the behavior of problems across a range of design parameters and operating conditions. However, the only drawbacks of *a priori* method are the need to use more complex meta-heuristics and to address conflicting objectives. Approximately the majority of the most well-known heuristic algorithms have been extended to solve multi-objective problem. In the following paragraphs the most popular and recent ones are briefly presented.

The literature shows that the most popular multi-objective meta-heuristic is Non-dominated Sorting GA (NSGA-II) (Deb et al., 2002), which is a multi-objective version of the well-regarded GA algorithm (Goldberg, 1989; Goldberg & Holland, 1988). This algorithm was proposed to alleviate the three problems of the first version (Srinivas & Deb, 1994). These problems are high computational cost of non-dominated sorting, lack of considering elitism, and lack of a sharing parameter (different from niching). In order to alleviate the aforementioned problems, NSGA-II utilizes a fast non-dominated sorting technique, an elitist-keeping technique, and a new niching operator which is parameter less.

The NSGA-II algorithm starts with a random population. The individuals are grouped based on the non-dominated sorting method. The fitness of each individual is defined based on its non-domination level. The second population is created by the selection, recombination, and mutation operators. Both populations create a new big population. This new population is then sorted again by the non-dominated sorting approach. The higher the non-domination level, the higher priority to select as a new individual for the final population. The process of selecting the non-dominated individuals should be repeated until having a population with the same size of the initial population. Finally, these steps are run until the satisfaction of an end criterion.

The second most popular multi-objective meta-heuristics is Multi-Objective Particle Swarm Optimization (MOPSO). The MOPSO algorithm was proposed by Coello et al., 2004; Coello Coello and Lechuga, 2002. Following the same concepts as PSO, it employs a number of particles, which fly around in the search space to find the best solution. Meanwhile, they all trace the best location (best solution) in their paths (Shi & Eberhart, 1998). In contrast to PSO, there is, of course, no single "best" solution to track. In other words, particles must consider their own non-dominated solutions (*pbest*) as well as one of the non-dominated solutions the swarm has obtained so far (*gbest*) when updating position. An external archive is used commonly for storing and retrieving the Pareto-optimal solutions obtained. In addition, a mutation operation called turbulence is embedded in MOPSO occasionally to increase randomness and promote diversity of trial solutions. A comprehensive survey of the PSO-based multi-objective optimizers can be found in Reyes-Sierra and Coello, 2006.

The proposed external archive is similar to the adaptive grid in Pareto Archived Evolution Strategy (PAES) (Knowles & Corne, 2000) as it has been designed to save the non-dominated solutions obtained so far. It has two main components: an archive controller and a grid. The former component is responsible for deciding if a solution should be added to the archive or not. If a new solution is dominated by one of the archive members it should be omitted immediately. If the new solution is not dominated by the archive members, it should be added to the archive. If a member of the archive is dominated by a new solution, it has to be replaced by the new solution. Finally, if the archive is full the adaptive grid mechanism is triggered.

The grid component is responsible for keeping the archive solutions as diverse as possible. In this method the objective space is divided into several regions. If a newly obtained solution lies outside the grid, all the grid locations should be recalculated to cover it. If a new solution lies within the grid, it is directed to the portion of the grid with the lowest number of particles. The main advantage of this grid is the low computational cost compared to niching (in worst case it is the same as niching $O(N^2)$ when the grid must be updated in each iteration).

MOPSO has a very fast convergence speed which could make it prone to premature termination with a false Pareto optimal front in multi-objective optimization (Nebro, Durillo, & Coello, 2013). The mutation strategy is helpful in this case. It randomly affects not only the particles in the swarm but also the design variables. The mutation rate is decreased over the course of iterations.

The MOPSO algorithm starts by randomly placing the particles in a problem space. Over the course of iterations, the velocities of particles are calculated. After defining the velocities, the position of particles can be updated. All the non-dominated solutions are added to the archive. Finally, the search process is terminated by satisfaction of a stopping criterion.

Over the past three years, many multi-objective optimization algorithms have been proposed: Multi-Objective Cat Swarm Optimization (MOCSO) (Pradhan & Panda, 2012), Multi-objective Ant Colony Optimization (Shi & Kong, 2015), Multi-objective Teaching–Learning-Based Optimization algorithm (Lin et al., 2015), Multi-objective Artificial Bee Colony algorithm (Hancer, Xue, Zhang, Karaboga, & Akay, 2015), Multi Objective Differential Evolution (Osorio Velazquez, Coello Coello, & Arias-Montano, 2014), and Multi-objective Gravitational Search Algorithm (MOGSA) (Hemmatian, Fereidoon, & Assareh, 2014). These studies show the ability of meta-heuristics in handling multiple objectives. Although all the above discussed algorithms are able to approximate the true Pareto optimal front of a given problem, they are not able to solve all optimization problems according to NFL theorem. Therefore, it is very likely that a new algorithm solve a problem that cannot be solved by the existing techniques in the literature. In the following section a novel multi-objective version of GWO is

proposed as an alternative to the current algorithms in the literature for solving multi-objective optimization problems.

3. Multi-Objective Grey Wolf Optimizer (MOGWO)

The GWO algorithm was proposed by Mirjalili, Mirjalili, and Lewis, 2014. The social leadership and hunting technique of grey wolves were the main inspiration of this algorithm. In order to mathematical model the social hierarchy of wolves when designing GWO, the fittest solution is considered as the alpha (α) wolf. Consequently, the second and third best solutions are named beta (β) and delta (δ) wolves, respectively. The rest of the candidate solutions are assumed to be omega (ω) wolves. In the GWO algorithm the hunting (optimization) is guided by α, β, and δ. The ω wolves follow these three wolves in the search for the global optimum.

In addition to the social leadership, the following equations were proposed in order to simulate the encircling behavior of grey wolves during hunt (Fig. 3) (Mirjalili et al., 2014):

$$\vec{D} = \left|\vec{C} \cdot \vec{X_p}(t) - \vec{X}(t)\right| \tag{3.1}$$

$$\vec{X}(t+1) = \vec{X_p}(t) - \vec{A} \cdot \vec{D} \tag{3.2}$$

where t indicates the current iteration, $\vec{A}$ and $\vec{C}$ are coefficient vectors, $\vec{X}_p$ is the position vector of the prey, and $\vec{X}$ indicates the position vector of a grey wolf.

The vectors $\vec{A}$ and $\vec{C}$ are calculated as follows:

$$\vec{A} = 2\vec{a} \cdot \vec{r}_1 - \vec{a} \tag{3.3}$$

$$\vec{C} = 2 \cdot \vec{r_2} \tag{3.4}$$

where elements of $\vec{a}$ linearly decrease from 2 to 0 over the course of iterations and r_1, r_2 are random vectors in [0,1]. Fig. 1

The GWO algorithm utilizes the simulated social leadership and encircling mechanism in order to find the optimal solution for optimization problems. This algorithm saves the first three best solutions obtained so far and obliges other search agents (including the omegas) to update their positions with respect to them. The following formulas are run constantly for each search agent during optimization in order to simulate the hunting and find promising regions of the search space:

$$\vec{D}_\alpha = \left|\vec{C}_1 \cdot \vec{X}_\alpha - \vec{X}\right| \tag{3.5}$$

$$\vec{D}_\beta = \left|\vec{C}_2 \cdot \vec{X}_\beta - \vec{X}\right| \tag{3.6}$$

$$\vec{D}_\delta = \left|\vec{C}_3 \cdot \vec{X}_\delta - \vec{X}\right| \tag{3.7}$$

$$\vec{X}_1 = \vec{X}_\alpha - \vec{A}_1 \cdot \left(\vec{D}_\alpha\right) \tag{3.8}$$

$$\vec{X}_2 = \vec{X}_\beta - \vec{A}_2 \cdot \left(\vec{D}_\beta\right) \tag{3.9}$$

$$\vec{X}_3 = \vec{X}_\delta - \vec{A}_3 \cdot \left(\vec{D}_\delta\right) \tag{3.10}$$

$$\vec{X}(t+1) = \frac{\vec{X}_1 + \vec{X}_2 + \vec{X}_3}{3} \tag{3.11}$$

The exploration is guaranteed by $\vec{A}$ with random values greater than 1 or less than –1 that obliges the search agent to diverge from the prey. Another component of GWO that favors exploration is $\vec{C}$. The $\vec{C}$ vector generates random values in [0, 2], in which random weights for prey are provided in order to stochastically emphasize ($C>1$) or deemphasize ($C<1$) the effect of prey in defining the distance in Eq. (3.1). This assists GWO to show a more random behavior throughout optimization, favoring exploration and local optima avoidance. It is worth mentioning here that C is not linearly decreased in contrast to A. The C parameter was deliberately required to provide random values at all times in order to emphasize exploration not only during initial iterations but also final iterations. This component is very helpful in case of local optima stagnation, especially in the final iterations.

The exploitation of the GWO algorithm starts when $|A|<1$. When random values of $\vec{A}$ are in [–1,1], the next position of a search agent can be in any position between its current position and the position of the prey, which assists the search agents to converge towards an estimated position of prey provided by alpha, beta, and delta solutions.

The GWO algorithm starts optimization with generating a set of random solutions as the first population. During optimization, the three best obtained solutions so far are saved and considered as alpha, beta, and delta solutions. For every omega wolf (search agents except α, β, and δ), the position updating Formula (3.5) to (3.11) are triggered. Meanwhile, parameters a and A are linearly decreased over the course of iteration. Therefore, Search agents tend to diverge from the prey when $|\vec{A}|>1$ and converge towards the prey when $|\vec{A}|<1$. Finally, the position and score of the alpha solution is returned as the best solutions obtained throughout optimization when an end condition is satisfied.

In order to perform multi-objective optimization by GWO, we integrate two new components. The employed components are very similar to those of MOPSO (Coello et al., 2004). The first one is an archive, which is responsible for storing non-dominated Pareto optimal solutions obtained so far. The second component is a leader selection strategy that assists to choose alpha, beta, and delta solutions as the leaders of the hunting process from the archive.

The archive is a simple storage unit that can save or retrieve non-dominated Pareto optimal solutions obtained so far. The key module of the archive is an archive controller, which controls the archive when a solution wants to enter the archive or when the archive is full. Note that there is a maximum number of members for the archive. During the course of iteration, non-dominated solutions obtained so far are compared against the archive residents. There would be three different possible cases as follows:

- The new member is dominated by at least one of the archive residences. In this case the solution should not be allowed to enter the archive.
- The new solution dominates one or more solutions in the archive. In this case the dominated solution(s) in the archive should be omitted and the new solution will be able to enter the archive.
- If neither the new solution nor archive members dominate each other, the new solution should be added to the archive.
- If the archive is full, the grid mechanism should be first run to re-arrange the segmentation of the objective space and find the most crowded segment to omit one of its solutions. Then, the new solution should be inserted to the least crowded segment in order to improve the diversity of the final approximated Pareto optimal front.

The probability of deleting a solution is increased proportional to the number of solutions in the hypercube (segment). For removing solutions if the archive was full, the most crowded segments are first selected, and a solution is omitted from one of them randomly in order to provide a space for the new solution. There is a special case where a solution is inserted outside the hypercubes. In this case, all the segments are extended in order to cover the new solutions. So, the segments of other solutions can be changed as well.

The second component is the leader selection mechanism. In GWO, three of the best solutions obtained so far are used as alpha, beta, and delta wolves. These leaders guide the other search agents

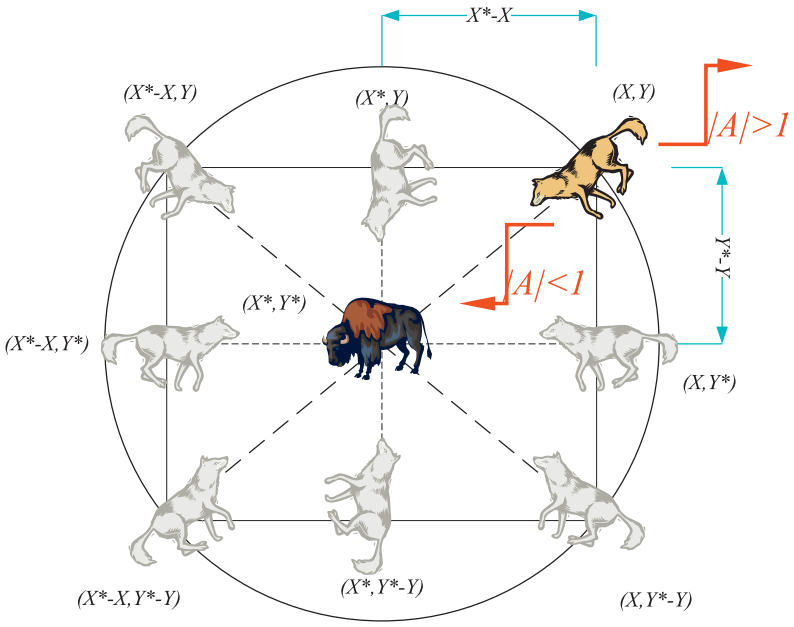

Fig. 1. Position updating mechanism of search agents and effects of *A* on it.

toward promising regions of the search space with the hope to find a solution close to the global optimum. In a multi-objective search space, however, the solutions cannot easily be compared due to the Pareto optimality concepts as discussed in the preceding subsection. The leader selection mechanism is designed to handle this issue. As mentioned above there is an archive of the best non-dominated solutions obtained so far. The leader selection component chooses the least crowded segments of the search space and offers one of its non-dominated solutions as alpha, beta, or delta wolves. The selection is done by a roulette-wheel method with the following probability for each hypercube:

$$P_i = \frac{c}{N_i} \tag{3.12}$$

where c is a constant number greater than one and N is the number of obtained Pareto optimal solutions in the ith segment.

It may be seen in Eq. (3.11) that less crowded hypercubes have higher probability of suggesting new leaders. The probability of choosing a hypercube to select leaders from is increased when the number of obtained solutions is decreased in the hypercube. It should be noted that there might be some special cases since we have to choose three leaders. If there are three solutions in the least crowded segment, three of them are randomly assigned to alpha, beta, and delta solutions. If there are less than three solutions in the least crowded hypercube, the second least crowded hypercube is also found to choose other leaders from. This scenario is the same if the second least crowded hypercube has one solution, so the delta leader should be chosen from the third least crowded hypercube. With this method, we prevent MOGWO from picking similar leaders for alpha, beta, or delta. Consequently, the search is always toward the unexplored/unexposed areas of the search space since the leader selection mechanism favors the least crowded hypercubes and offers leaders from different segments if there is not enough number of leaders (less than 3) in the least crowded segment.

The computational complexity of MOGWO is of $O(MN^2)$ where N is the number of individuals in the population and M is the number of objectives. The complexity is equal to other well-known algorithms in this field: NSGA-II (Deb et al., 2002), MOPSO, SPEA2 (Zitzler et al., 2001), and PAES (Knowles & Corne, 1999). However, the computational complexity is better than some of the algorithms such as NSGA (Srinivas & Deb, 1994) and SPEA (Zitzler & Thiele, 1998), which are of $O(MN^3)$.

After all, the pseudo codes of the MOGWO algorithm are provided in Fig. 2.

To see how the proposed MOGWO algorithm can be theoretically effective in solving multi-objective problems some remarks may be noted as follows:

- The employed external archive effectively saves the best non-dominated solutions obtained so far.
- Since MOGWO inherits the encircling mechanism of GWO, there is a circle-shaped neighborhood around the solutions which can be extended to higher dimensions as a hyper-sphere (in the parameter space).
- The random parameters A and C assist candidate solutions to have hyper-spheres with different random radii.
- Since MOGWO inherits the hunting mechanism of GWO, the search agents are allowed to locate the probable position of the prey.
- Exploration and exploitation are guaranteed by the adaptive values of a and A.
- The adaptive values of parameters a and A allow MOGWO to smoothly transition between exploration and exploitation. Therefore, the convergence of the MOGWO algorithm is guaranteed.
- With decreasing A, half of the iterations are devoted to exploration ($|A| \geq 1$) and the other half is dedicated to exploitation ($|A| < 1$).
- The MOGWO has only two key parameters to be adjusted (a and C).
- The grid mechanism and selection leader component maintain the diversity of the archive during optimization.
- Employed roulette-wheel in the leader selection component provides a low probability to choose leaders from most crowded hypercubes as well. This emphasizes local front avoidance of MOGWO.
- Non-adaptive random values for C parameter during optimization also boost exploration and the local front avoidance of the MOGWO algorithm simultaneously.

It should be noted here that the convergence of the MOGWO algorithm is guaranteed because it utilizes the same mathematical model to search for optimal solutions. It has been proved that

Initialize the grey wolf population X_i *(i = 1, 2, ..., n)*
Initialize a, A, and C
Calculate the objective values for each search agent
Find the non-dominated solutions and initialized the archive with them
X_α*=SelectLeader(archive)*
Exclude alpha from the archive temporarily to avoid selecting the same leader
X_β*= SelectLeader(archive)*
Exclude beta from the archive temporarily to avoid selecting the same leader
X_δ*= SelectLeader(archive)*
Add back alpha and beta to the archive
t=1;
while *(t < Max number of iterations)*
  for *each search agent*
    Update the position of the current search agent by equations (3.5)-(3.11)
  end for
  Update a, A, and C
  Calculate the objective values of all search agents
  Find the non-dominated solutions
  Update the archive with respect to the obtained non-dominated solutions
  If *the archive is full*
    Run the grid mechanism to omit one of the current archive members
    Add the new solution to the archive
  end if
  If *any of the new added solutions to the archive is located outside the hypercubes*
    Update the grids to cover the new solution(s)
  end if
  X_α*=SelectLeader(archive)*
  Exclude alpha from the archive temporarily to avoid selecting the same leader
  X_β*= SelectLeader(archive)*
  Exclude beta from the archive temporarily to avoid selecting the same leader
  X_δ*= SelectLeader(archive)*
  Add back alpha and beta to the archive
  t=t+1
end while
return ***archive***

Fig. 2. Pseudo code of the MOGWO algorithm.

GWO requires the search agents to change the positions abruptly in the initial steps of optimization and gradually in the final steps (Mirjalili et al., 2014). According to Van den Bergh and Engelbrecht, 2006, this behavior guarantees the convergence of an algorithm in the search space. The MOGWO algorithm inherits all the characteristics of GWO, which means that the search agents explore and exploit the search space in a same manner. The main difference is that MOGWO searches around a set of archive members (which might be different even if the archive does not change), while GWO only saves and improves three best solutions.

The difference between the proposed algorithm and one of its most recent counterparts, Multi-Objective Cat Swarm Optimization (MOCSO) proposed by Pradhan and Panda, 2012, is that MOCSO employs both non-dominated sorting and archive, whereas MOGWO only utilises an external archive to maintain the non-dominated solutions.

4. Results and discussion

This section outlines experimental setup, presents results, and provides discussion.

4.1. Experimental setup

The MOGWO algorithm is compared to two well-known algorithms in the literature: MOPSO and MOEA/D. For MOPSO, the following initial parameters are chosen:

- $\phi_1 = \phi_2 = 2.05$
- $\phi = \phi_1 + \phi_2$
- $w = \chi = \frac{2}{\phi - 2 + \sqrt{\phi^2 - 4\phi}}$: inertia weight
- $c_1 = \chi * \phi_1$: personal coefficient
- $c_2 = \chi * \phi_2$: social coefficient
- $\alpha = 0.1$: grid inflation parameter
- $\beta = 4$: leader selection pressure parameter
- $nGrid = 10$: number of grids per each dimension

For MOEA/D, the following initial parameters are chosen:

- N= 100: Subproblems
- $T = 0.1N = 10$: number of neighbors
- $n_r = 0.01N = 1$: the maximal copies of a new child in update
- $\delta = 0.9$: the probability of selecting parents from the neighborhood
- $CR = F = 0.5$: mutation rates
- $\eta = 30$: distribution index

Note that for all of the experiments, we have utilized 100 search agents and a maximum of 3000 iterations. As test beds, we chose 10 standard multi-objective test problems proposed in CEC 2009 (Zhang et al., 2008). The benchmark problems are provided in Tables 1 and 2. These test problems are considered as the most challenging test problems in the literature that provide different multi-objective search spaces with different Pareto optimal fronts: convex, non-convex, discontinuous, and multi-modal.

For the performance metric (Fonseca, Knowles, Thiele, & Zitzler, 2005), we have used Inverted Generational Distance (IGD) for measuring convergence. The Spacing (SP) (Carlos A Coello Coello et al., 2004; Schott, 1995) and Maximum Spread (MS) (Zitzler, 1999) are employed to quantify and measure the coverage. The mathematical formulation of IGD is similar to that of Generational Distance (GD) (Van Veldhuizen & Lamont, 1998). This modified measure was pro-

Table 1
Bi-objective test problems.

Name	Mathematical formulation
UF1	$f_1 = x_1 + \frac{2}{\lvert J_1\rvert}\sum_{j\in J_1}[x_j - \sin(6\pi x_1 + \frac{j\pi}{n})]^2,\ f_2 = 1-\sqrt{x} + \frac{2}{\lvert J_2\rvert}\sum_{j\in J_2}[x_j - \sin(6\pi x_1 + \frac{j\pi}{n})]^2$
	$J_1 = \{j \mid j \text{ is odd and } 2 \le j \le n\},\ J_2 = \{j \mid j \text{ is even and } 2 \le j \le n\}$
UF2	$f_1 = x_1 + \frac{2}{\lvert J_1\rvert}\sum_{j\in J_1} {y_j}^2,\ f_2 = 1-\sqrt{x} + \frac{2}{\lvert J_2\rvert}\sum_{j\in J_2} {y_j}^2$
	$J_1 = \{j \mid j \text{ is odd and } 2 \le j \le n\},\ J_2 = \{j \mid j \text{ is even and } 2 \le j \le n\}$
	$y_j = \begin{cases} x_j - [0.3x_1^2\cos(24\pi x_1 + \frac{4j\pi}{n}) + 0.6x_1]\cos(6\pi x_1 + \frac{j\pi}{n}) & \text{if } j \in J_1 \\ x_j - [0.3x_1^2\cos(24\pi x_1 + \frac{4j\pi}{n}) + 0.6x_1]\sin(6\pi x_1 + \frac{j\pi}{n}) & \text{if } j \in J_2 \end{cases}$
UF3	$f_1 = x_1 + \frac{2}{\lvert J_1\rvert}(4\sum_{j\in J_1} y_j^2 - 2\prod_{j\in J_1}\cos(\frac{20y_j\pi}{\sqrt{j}}) + 2) f_2 = \sqrt{x_1} + \frac{2}{\lvert J_2\rvert}(4\sum_{j\in J_1} y_j^2 - 2\prod_{j\in J_2}\cos(\frac{20y_j\pi}{\sqrt{j}}) + 2)$
	$J_1 \text{and } J_2 \text{are the same as those of } UF1,\ y_j = x_j - x_1^{0.5(1.0+\frac{3(j-2)}{n-2})},\ j = 2,3,\ldots,n$
UF4	$f_1 = x_1 + \frac{2}{\lvert J_1\rvert}\sum_{j\in J_1} h(y_j),\ f_2 = 1 - x_2 + \frac{2}{\lvert J_2\rvert}\sum_{j\in J_2} h(y_j)$
	$J_1 \text{and } J_2 \text{are the same as those of } UF1,\ y_j = x_j - \sin(6\pi x_1 + \frac{j\pi}{n}),\ j = 2,3,\ldots,n,\ h(t) = \frac{\lvert t\rvert}{1+e^{2\lvert t\rvert}}$
UF5	$f_1 = x_1 + (\frac{1}{2N} + \epsilon)\lvert\sin(2N\pi x_1)\rvert + \frac{2}{\lvert J_1\rvert}\sum_{j\in J_1} h(y_i),\ f_1 = 1 - x_1 + (\frac{1}{2N} + \epsilon)\lvert\sin(2N\pi x_1)\rvert + \frac{2}{\lvert J_2\rvert}\sum_{j\in J_2} h(y_i)$
	$J_1 \text{amd } J_2 \text{are identical to those of } UF1,\ \epsilon > 0,\ y_j = x_j - \sin(6\pi x_1 + \frac{j\pi}{n}),\ j = 2,3,\ldots,n$
	$h(t) = 2t^2 - \cos(4\pi t) + 1$
UF6	$f_1 = x_1 + \max\{0, 2(\frac{1}{2N} + \epsilon)\sin(2N\pi x_1)\} + \frac{2}{\lvert J_1\rvert}(4\sum_{j\in J_1} y_j^2 - 2\prod_{j\in J_1}\cos(\frac{20y_j\pi}{\sqrt{j}}) + 1))$
	$f_2 = 1 - x_1 + \max\{0, 2(\frac{1}{2N} + \epsilon)\sin(2N\pi x_1)\} \frac{2}{\lvert J_2\rvert}(4\sum_{j\in J_2} y_j^2 - 2\prod_{j\in J_2}\cos(\frac{20y_j\pi}{\sqrt{j}}) + 1))$
	$J_1 \text{amd } J_2 \text{are identical to those of } UF1,\ \epsilon > 0,\ y_j = x_j - \sin(6\pi x_1 + \frac{j\pi}{n}),\ j = 2,3,\ldots,n$
UF7	$f_1 = \sqrt[5]{x_1} + \frac{2}{\lvert J_1\rvert}\sum_{j\in J_1} y_j^2,\ f_2 = 1 - \sqrt[5]{x_1} + \frac{2}{\lvert J_2\rvert}\sum_{j\in J_2} y_j^2$
	$J_1 \text{amd } J_2 \text{are identical to those of } UF1,\ \epsilon > 0,\ y_j = x_j - \sin(6\pi x_1 + \frac{j\pi}{n}),\ j = 2,3,\ldots,n$

Table 2
Tri-objective test problems.

Name	Mathematical formulation
UF8	$f_1 = \cos(0.5x_1\pi)\cos(0.5x_2\pi) + \frac{2}{\lvert J_1\rvert}\sum_{j\in J_1}(x_j - 2x_2\sin(2\pi x_1 + \frac{j\pi}{n})^2)$
	$f_2 = \cos(0.5x_1\pi)\sin(0.5x_2\pi) + \frac{2}{\lvert J_2\rvert}\sum_{j\in J_2}(x_j - 2x_2\sin(2\pi x_1 + \frac{j\pi}{n})^2)$
	$f_3 = \sin(0.5x_1\pi) + \frac{2}{\lvert J_3\rvert}\sum_{j\in J_3}(x_j - 2x_2\sin(2\pi x_1 + \frac{j\pi}{n})^2)$
	$J_1 = \{j \mid 3 \le j \le n,\ \text{and } j-1 \text{ is a multiplication of } 3\},\ J_2 = \{j \mid 3 \le j \le n,\ \text{and } j-2 \text{ is a multiplication of } 3\},$
	$J_3 = \{j \mid 3 \le j \le n,\ \text{and } j \text{ is a multiplication of } 3\},$
UF9	$f_1 = 0.5[\max\{0, (1+\epsilon)(1-4(2x_1-1)^2)\} + 2x_1]x_2 + \frac{2}{\lvert J_1\rvert}\sum_{j\in J_1}(x_j - 2x_2\sin(2\pi x_1 + \frac{j\pi}{n})^2)$
	$f_2 = 0.5[\max\{0, (1+\epsilon)(1-4(2x_1-1)^2)\} + 2x_1]x_2 + \frac{2}{\lvert J_2\rvert}\sum_{j\in J_2}(x_j - 2x_2\sin(2\pi x_1 + \frac{j\pi}{n})^2)$
	$f_3 = 1 - x_2 + \frac{2}{\lvert J_3\rvert}\sum_{j\in J_3}(x_j - 2x_2\sin(2\pi x_1 + \frac{j\pi}{n})^2)$
	$J_1 = \{j \mid 3 \le j \le n,\ \text{and } j-1 \text{ is a multiplication of } 3\},\ J_2 = \{j \mid 3 \le j \le n,\ \text{and } j-2 \text{ is a multiplication of } 3\},$
	$J_3 = \{j \mid 3 \le j \le n,\ \text{and } j \text{ is a multiplication of } 3\},\ \epsilon = 0.1$
UF10	$f_1 = \cos(0.5x_1\pi)\cos(0.5x_2\pi) + \frac{2}{\lvert J_1\rvert}\sum_{j\in J_1}[4y_j^2 - \cos(8\pi y_j) + 1]$
	$f_2 = \cos(0.5x_1\pi)\sin(0.5x_2\pi) + \frac{2}{\lvert J_2\rvert}\sum_{j\in J_1}[4y_j^2 - \cos(8\pi y_j) + 1]$
	$f_3 = \sin(0.5x_1\pi) + \frac{2}{\lvert J_3\rvert}\sum_{j\in J_1}[4y_j^2 - \cos(8\pi y_j) + 1]$
	$J_1 = \{j \mid 3 \le j \le n,\ \text{and } j-1 \text{ is a multiplication of } 3\},\ J_2 = \{j \mid 3 \le j \le n,\ \text{and } j-2 \text{ is a multiplication of } 3\},$
	$J_3 = \{j \mid 3 \le j \le n,\ \text{and } j \text{ is a multiplication of } 3\},$

posed by Sierra and Coello, 2005 and formulated as follows:

$$\text{IGD} = \frac{\sqrt{\sum_{i=1}^{n} d_i^2}}{n} \tag{4.1}$$

where n is the number of true Pareto optimal solutions and d_i indicates the Euclidean distance between the ith true Pareto optimal solution and the closest obtained Pareto optimal solutions in the reference set. The Euclidean distance between obtained solutions and reference set is different here. In IGD, the Euclidean distance is calculated for every true solution with respect to its nearest obtained Pareto optimal solutions in the objective space.

The mathematical formulation of the SP and MS measures are as follows:

$$\text{SP} = \sqrt{\frac{1}{n-1}\sum_{i=1}^{n}\left(\bar{d} - d_i\right)^2} \tag{4.2}$$

where $\bar{d}$ is the average of all d_i, n is the number of Pareto optimal solutions obtained, and $d_i = \min_j(|f_1^i(\vec{x}) - f_1^j(\vec{x})| + |f_2^i(\vec{x}) - f_2^j(\vec{x})|)$ for all i, j=1,2,3,…,n.

$$\text{MS} = \sqrt{\sum_{i=1}^{o}\max\left(d(a_i, b_i)\right)} \tag{4.3}$$

where d is a function to calculate the Euclidean distance, a_i is the maximum value in the ith objective, b_i is the minimum in the ith objective, and o is the number of objectives.

In addition to utilizing the above performance metrics, which allow us to quantitatively compare MOGWO with MOPSO and MOEA/D, we illustrate the best set of Pareto optimal solutions obtained by each algorithm on both parameter space and search space. This allows us to compare the performance of the algorithms qualitatively as well. All the algorithms are run 10 times on the test problems and the statistics results of these 10 runs are provided in Tables 3–5. Note

Table 3.
Statistical results for IGD on UF1 to UF10.

	UF1 (bi-objective)			*UF2 (bi-objective)*			*UF3 (bi-objective)*		
IGD	MOGWO	MOPSO	MOEA/D	MOGWO	MOPSO	MOEA/D	MOGWO	MOPSO	MOEA/D
Average	0.114425	0.137005	0.187135	0.05825	0.060405	0.122343	0.255691	0.313999	0.288648
Median	0.113	0.131745	0.182855	0.057775	0.04835	0.120135	0.25091	0.30802	0.289295
STD. Dev.	0.019538	0.044068	0.05073	0.007392	0.027625	0.0107	0.080702	0.044726	0.015921
Worst	0.15774	0.22786	0.24642	0.07322	0.13051	0.14369	0.36786	0.37773	0.31294
Best	0.08023	0.0899	0.12652	0.0498	0.03699	0.10486	0.1295	0.25648	0.26342
	UF4 (bi-objective)			*UF5 (bi-objective)*			*UF6 (bi-objective)*		
IGD	MOGWO	MOPSO	MOEA/D	MOGWO	MOPSO	MOEA/D	MOGWO	MOPSO	MOEA/D
Average	0.058669	0.136037	0.068131	0.797072	2.202376	1.291451	0.279375	0.64752	0.688119
Median	0.058685	0.13432	0.06846	0.69942	2.125745	1.33761	0.24435	0.55073	0.698415
STD. Dev.	0.000481	0.007391	0.002143	0.378579	0.553042	0.134897	0.104485	0.266117	0.055326
Worst	0.05936	0.15189	0.07037	1.73857	3.03836	1.46746	0.55036	1.24281	0.74011
Best	0.05797	0.12733	0.06466	0.46795	1.46479	1.12306	0.19338	0.37933	0.55235
	UF7 (bi-objective)			*UF8 (tri-objective)*		*UF9 (tri-objective)*		*UF10 (tri-objective)*	
IGD	MOGWO	MOPSO	MOEA/D	MOGWO	MOPSO	MOGWO	MOPSO	MOGWO	MOPSO
Average	0.160359	0.353954	0.455242	2.057772	0.536709	0.191747	0.488503	3.594533	1.637196
Median	0.07336	0.387305	0.437665	2.335965	0.5364	0.166025	0.41451	2.82552	1.59163
STD. Dev.	0.139111	0.204421	0.189831	1.145524	0.182571	0.092504	0.144497	3.488293	0.298794
Worst	0.40142	0.61512	0.67701	3.87888	0.79637	0.44794	0.7221	12.95643	2.1622
Best	0.06275	0.05402	0.029	0.46131	0.2453	0.1291	0.33355	1.0431x4	1.22008

Table 4
Statistical results for SP on UF1 to UF10.

	UF1 (bi-objective)			*UF2 (bi-objective)*			*UF3 (bi-objective)*		
IGD	MOGWO	MOPSO	MOEA/D	MOGWO	MOPSO	MOEA/D	MOGWO	MOPSO	MOEA/D
Average	0.01237	0.00898	0.00384	0.01108	0.00829	0.00876	0.04590	0.00699	0.02680
Median	0.00536	0.00855	0.00382	0.00946	0.00814	0.00859	0.04860	0.00677	0.02505
STD. Dev.	0.01462	0.00247	0.00151	0.00362	0.00168	0.00076	0.01453	0.00170	0.02064
Worst	0.04641	0.01464	0.00665	0.01816	0.01245	0.01042	0.07050	0.01007	0.06256
Best	0.00081	0.00670	0.00213	0.00758	0.00624	0.00797	0.01549	0.00476	0.00078
	UF4 (bi-objective)			*UF5 (bi-objective)*			*UF6 (bi-objective)*		
IGD	MOGWO	MOPSO	MOEA/D	MOGWO	MOPSO	MOEA/D	MOGWO	MOPSO	MOEA/D
Average	0.00969	0.00666	0.00730	0.15231	0.00479	0.00278	0.01446	0.02084	0.00630
Median	0.00857	0.00662	0.00728	0.08778	0.00487	0.00007	0.01110	0.01235	0.00000
STD. Dev.	0.00390	0.00091	0.00059	0.16247	0.00408	0.00553	0.01246	0.03258	0.01267
Worst	0.01722	0.00809	0.00836	0.51247	0.01206	0.01615	0.04112	0.11140	0.03030
Best	0.00583	0.00546	0.00610	0.00843	0.00006	0.00000	0.00191	0.00215	0.00000
	UF7 (bi-objective)			*UF8 (tri-objective)*		*UF9 (tri-objective)*		*UF10 (tri-objective)*	
IGD	MOGWO	MOPSO	MOEA/D	MOGWO	MOPSO	MOGWO	MOPSO	MOGWO	MOPSO
Average	0.00824	0.00670	0.00540	0.00687	0.02682	0.01743	0.02343	0.02523	0.01994
Median	0.00548	0.00655	0.00441	0.00470	0.02639	0.01826	0.02349	0.02392	0.02066
STD. Dev.	0.00856	0.00285	0.00301	0.00474	0.00827	0.00633	0.00405	0.01500	0.00348
Worst	0.03106	0.01240	0.01168	0.01879	0.04473	0.02856	0.03087	0.05384	0.02665
Best	0.00031	0.00325	0.00084	0.00365	0.01531	0.00653	0.01716	0.00000	0.01536

that we used 300,000 function evaluation for each algorithm and the number of parameters for each of the test functions are 30. The qualitative results are also provided in Figs. 3–5.

4.2. Discussion of the results

Table 3 provides statistical results of the algorithm for IGD. This table shows that the proposed MOGWO algorithm is able to provide the best results on all the statistical metrics for UF1. The significance of the results is also illustrated in Fig. 3. This figure shows that the boxplot of MOGWO is significantly lower and narrower than those of MOPSO and MOEA/D. IGD is the performance metric that shows the accuracy and convergence of an algorithm. So, it can be stated that the MOGWO algorithm is able to provide superior convergence on UF1.

The obtained Pareto optimal solution of each algorithm on UF1 are also depicted in Fig. 4. It may be seen in this figure that the convergence and coverage (see Tables 4 and 5) of MOGWO is better than others. Although there are discontinuities on the Pareto optimal front of MOGWO, the coverage of the whole front is broader than MOPSO and MOEA/D on this test function. The obtained Pareto optimal fronts of MOGWO and MOPSO are slightly similar. However, it seems that the Pareto optimal solutions of MOGWO are closer to the true Pareto optimal front and highly distributed along both objectives.

As per the results of the algorithms on UF2 in Table 3, MOGWO is able to outperform others in average. However, the best result is achieved by MOPSO. These results show the performance of the MOGWO algorithm is more stable than MOPSO. Since IGD is a good metric to benchmark the convergence of an algorithm, these results indicate that the MOGWO algorithm has a better convergence on this benchmark function as well. The significance of the results is illustrated in Fig. 3. The boxplot of the MOGWO is narrower than MOPSO and MOEA/D, showing stability of MOGWO algorithm in converging towards the true Pareto optimal front. The worst results belong to MOEA/D dealing with this benchmark problem. The boxplots of Fig. 3 shows that this algorithm provides very poor results.

In order to observe the coverage of the algorithms, the obtained Pareto optimal solutions are illustrated in Fig. 4. This figure shows that the obtained Pareto optimal front of MOGWO and MOEA/D are

Table 5
Statistical results for MS on UF1 to UF10.

	UF1 (bi-objective)			*UF2 (bi-objective)*			*UF3 (bi-objective)*		
IGD	MOGWO	MOPSO	MOEA/D	MOGWO	MOPSO	MOEA/D	MOGWO	MOPSO	MOEA/D
Average	0.92680	0.64538	0.51774	0.90972	0.91205	0.87201	0.94982	0.61030	0.23994
Median	0.93272	0.66322	0.59541	0.91041	0.91636	0.87437	1.00000	0.61612	0.22943
STD. Dev.	0.06884	0.19292	0.16609	0.02867	0.02560	0.00560	0.08777	0.10575	0.12129
Worst	0.81797	0.26592	0.31487	0.84695	0.86654	0.85986	0.76809	0.38172	0.08975
Best	0.99711	0.95226	0.74128	0.94791	0.95301	0.87794	1.00000	0.77145	0.47863
	UF4 (bi-objective)			*UF5 (bi-objective)*			*UF6 (bi-objective)*		
IGD	MOGWO	MOPSO	MOEA/D	MOGWO	MOPSO	MOEA/D	MOGWO	MOPSO	MOEA/D
Average	0.94242	0.81275	0.88320	0.39503	0.27926	0.29215	0.67360	0.27435	0.09677
Median	0.94269	0.81321	0.88131	0.43258	0.28654	0.29165	0.70826	0.22917	0.00005
STD. Dev.	0.00093	0.01367	0.01812	0.17494	0.09575	0.03470	0.12323	0.11285	0.20715
Worst	0.94095	0.79441	0.85324	0.03006	0.15574	0.23834	0.38838	0.15436	0.00000
Best	0.94327	0.83449	0.91394	0.61042	0.43827	0.34380	0.81492	0.52516	0.59484
	UF7 (bi-objective)			*UF8 (tri-objective)*		*UF9 (tri-objective)*		*UF10 (tri-objective)*	
IGD	MOGWO	MOPSO	MOEA/D	MOGWO	MOPSO	MOGWO	MOPSO	MOGWO	MOPSO
Average	0.80126	0.42928	0.56317	0.44573	0.50810	0.83991	0.19816	0.29721	0.13015
Median	0.96293	0.29520	0.63267	0.44429	0.50601	0.91055	0.16566	0.14238	0.10913
STD. Dev.	0.30865	0.27553	0.24209	0.18574	0.16136	0.19759	0.16351	0.34651	0.06263
Worst	0.02252	0.14458	0.14963	0.18863	0.22723	0.28750	0.06771	0.03194	0.06489
Best	0.98746	0.87714	0.99152	0.86376	0.71476	0.93753	0.64242	0.92828	0.25404

more distributed than MOEA/D. However, the convergence of the MOEA/D algorithm on the left side of the true Pareto optimal front is noticeable. Among the Pareto optimal fronts of MOGWO and MOPSO, that of MOPSO shows slightly better coverage as per the results of SP and MS in Tables 4 and 5.

The statistical results of the algorithm on UF3 for IGD show that MOGWO has the best convergence average. The superioir results can be observed on median and best results as well in Fig. 3. However, the best results for standard deviation and worst belong to MOEA/D. Fig. 4 shows a different behavior for the algorithms compared to the previous test problems. This figure proves that the superioirty of the MOGWO is not significant on UF3. However, the results of the MOGWO algorithm tend to be better than MOPSO and MOEA/D.

The true Pareto optimal front and best obtained Pareto optimal fronts for UF3 in Fig. 4 show that the Pareto optimal front of MOGWO is highly better than that of MOPSO and MOEA/D. For one, the MOGWO's front is more distributed than others. For another, the Pareto optimal solutions obtained are very close to the true Pareto optimal front especially on the right side. Another fact worth noticing here is the poor distribution of MOEA/D's obtained front, which shows that this algorithm is not able to show good coverage on UF3 benchmark problem. Despite close coverage of MOPSO (see Tables 3 and 4), the convergence of this algorithm is not as good as that of MOGWO.

The statistical results of MOGWO, MOPSO, and MOEA/D on UF4 benchmark problem show that MOGWO is able to outperform MOPSO and MOEA/D. According to Fig. 3, this superiority is very significant since the boxplot of MOGWO is super narrow and located under the minima of MOPSO and MOEA/D. These results prove that the proposed algorithm is able to provide remarkable convergence and coverage ability in solving multi-objective problems. Another fact worth mentioning here is the poor performance of MOPSO on this algorithm, which is not consistent to the results of the previous test problems. This might be due to the non-convex shape of the true Pareto optimal front, which is discussed in the next paragraph.

The shapes of the obtained Pareto optimal fronts of the algorithms provided in Fig. 4 shows that the shape of the true Pareto optimal front is convex, which is different from the previous test problems. This difference causes diverse shapes for obtained Pareto optimal front of each algorithm. The distributions of the obtained fronts are similar for all the algorithms to some extent, slightly better for the MOEA/D algorithm. However, the Pareto optimal solutions of MOGWO are closer to the true Pareto optimal front compared to MOPSO and MOEA/D. Despite these close results on the best obtained Pareto optimal front, quantitative results prove that the stability of results of the proposed MOGWO algorithm is significantly high on this benchmark problem.

The fifth test problem has a discontinuous Pareto optimal set and Pareto optimal front. The statistical results of the algorithms over 10 independent runs presented in tables show again that MOGWO provides the best performance dealing with UF5 test problem. This proves that this algorithm shows better convergence ability in average. However, the Pareto optimal solutions obtained in Fig. 4 and quantitative results in Tables 4 and 5 show poor convergence and coverage for all of algorithms. This might be due to the difficulties of the search space of UF5 with a huge number of discontinuous regions that prevents algorithms from providing accurate results on this test problem. We employed a 30 dimensional version of UF5, which make it more challenging for the algorithms. Although all algorithms show very poor performance, the MOGWO algorithm is significantly better as the results suggest.

Another test problem with discontinuous search space is UF6. A similar behavior to UF5 can be observed in the results. The statistical results for performance metrics show significant superior performance of MOGWO on UF6. The true Pareto optimal front of UF6 has three discontinuous parts separated from 0 to 0.2 and 0.5 to 0.7 over the first objective function ($f1$). None of the algorithms managed to obtain close Pareto optimal solutions to the true front. However, the results of MOGWO are better than MOPSO and MOEA/D. As may be seen in Fig. 5, it seems that MOGWO tend to find the three disconnected regions of the true Pareto optimal front, whereas MOPSO and MOEA/D only move towards one part. This proves that the proposed MOGWO algorithm has the potential to provide superior results on problems with discontinuities in their parameter/objective space. The results of Fig. 3 evidence that this superiority can be significant as well.

The UF7 test problem has a linear Pareto optimal front that makes it easier than problems with convex and non-convex shaped true Pareto optimal fronts. The statistical results for performance metrics suggest that the proposed algorithm shows better results. This superiority is not very significant which is due to linear shaped Pareto front of this test problem. MOGWO provides better results on average, median, standard deviation, and worst of IGD, SP, and MS. The obtained Pareto optimal solutions in Fig. 5 show that the best front

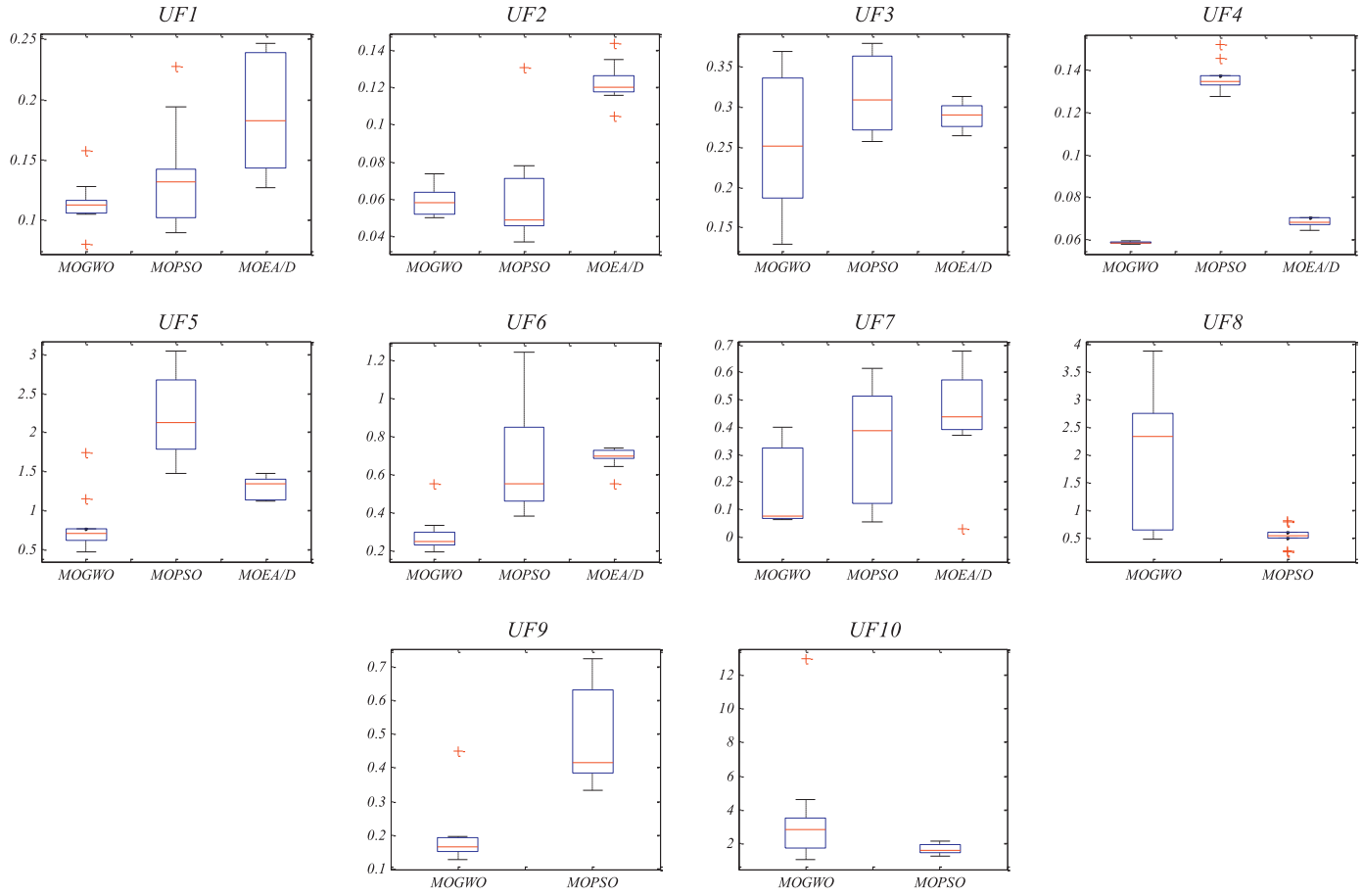

Fig. 3. Boxplot of the statistical results for IGD on UF1 to UF10.

of MOEA/D is uniformly distributed along the true Pareto optimal front. This is due to the nature of this algorithm, in which objectives are decomposed by a set of weights so linear Pareto optimal fronts can be easily constructed. The behavior of MOGWO and MOPSO are also almost similar. However, it may be observed in the quantitative results MOGWO is better in terms of improved convergence and coverage.

Problems UF8, UF9, and UF10 are three-objective, which make them more challenging than two-objective problems. Since three-objective version of MOEA/D has not been developed in Matlab, we compare MOGWO with just MOPSO for these benchmark problems. The results of the first three-objective problem show that MOPSO has higher IGD, SP, and MS statistical results than MOGWO. These results are supported by the box plots of the Fig. 3, in which the significance of the results can be observed.

The Pareto optimal fronts obtained by both algorithms after solving UF8 are shown in Fig. 5. It may be seen that the Pareto optimal solutions of MOPSO are far from the true Pareto optimal front. The Pareto optimal solutions obtained by MOGWO, however, are converged towards the true Pareto optimal font despite their low diversity. Therefore, the MOGWO algorithm shows high convergence but low coverage on UF8 in the best case.

The UF9 test problem has a separated Pareto front, making it more challenging. As per the results in the tables and figures, the proposed algorithm is again able to provide very promising results on this challenging test function. The statistical results are an evidence of the stable convergence of this algorithm over 10 independent runs. The quantitative and qualitative results also show that the coverage of MOGWO is very good along the objectives. Fig. 5 shows that MOPSO found very poor Pareto optimal solutions for this problem.

Finally, the results of the last problem, UF10 are similar to that of UF8, in which MOPSO provides better statistical results on average, median, standard deviation, and worst for IGD. MOGWO, however, provides the best coverage among 10 independent runs. It may be observed that the MOGWO algorithm shows very good coverage when solving UF10 as per the quantitative and qualitative results.

To have an image of the Pareto optimal sets obtained, the first three variables of the true Pareto optimal set and determined sets by MOGWO are illustrated in Fig. 6. The convergence and coverage of the MOGWO algorithms can be clearly seen in the Pareto optimal sets obtained in this figure.

The results and discussions evidence that the MOGWO algorithm is able to provide very competitive and promising results on the multi-objective test functions. The statistical results for IGD demonstrate the convergence ability of MOGWO. High convergence of MOGWO is originated from the concepts of attacking prey and updating position of omega wolves with respect to other wolves when $|A|<1$. When half of the iteration is passed, the search agents of MOGWO tend to exploit promising regions of the search space provided by alpha, beta, and delta solutions. Another observation was the high coverage of the obtained Pareto optimal solutions as per the results of SP and MS. This is due to the updating mechanism of MOGWO based on alpha, beta, and delta solutions, which allows it to consider three best solutions obtained so far and oblige omega wolves to update their positions with respect to them. In addition, we chose a new alpha, beta, and delta solutions for updating the position of each omega wolf, which reemphasize exploration.

It is worth discussing here that the main controlling parameter in GWO and MOGWO is A. We have utilized the same adaptive values in MOGWO as of the GWO algorithm. However, we have done

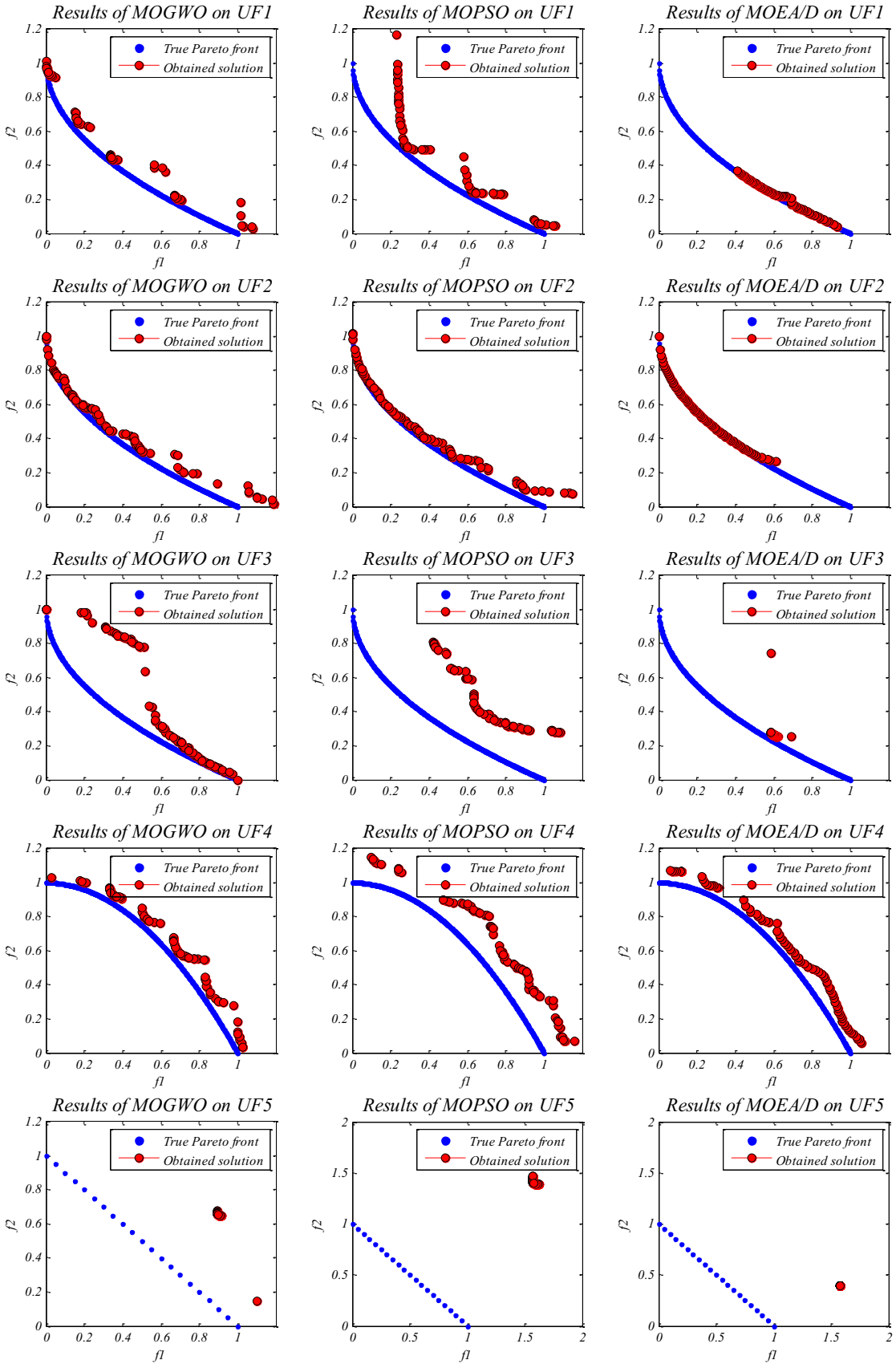

Fig. 4. Obtained Pareto optimal solutions by MOGWO, MOPSO, and MOEA/D for UF1 to UF5.

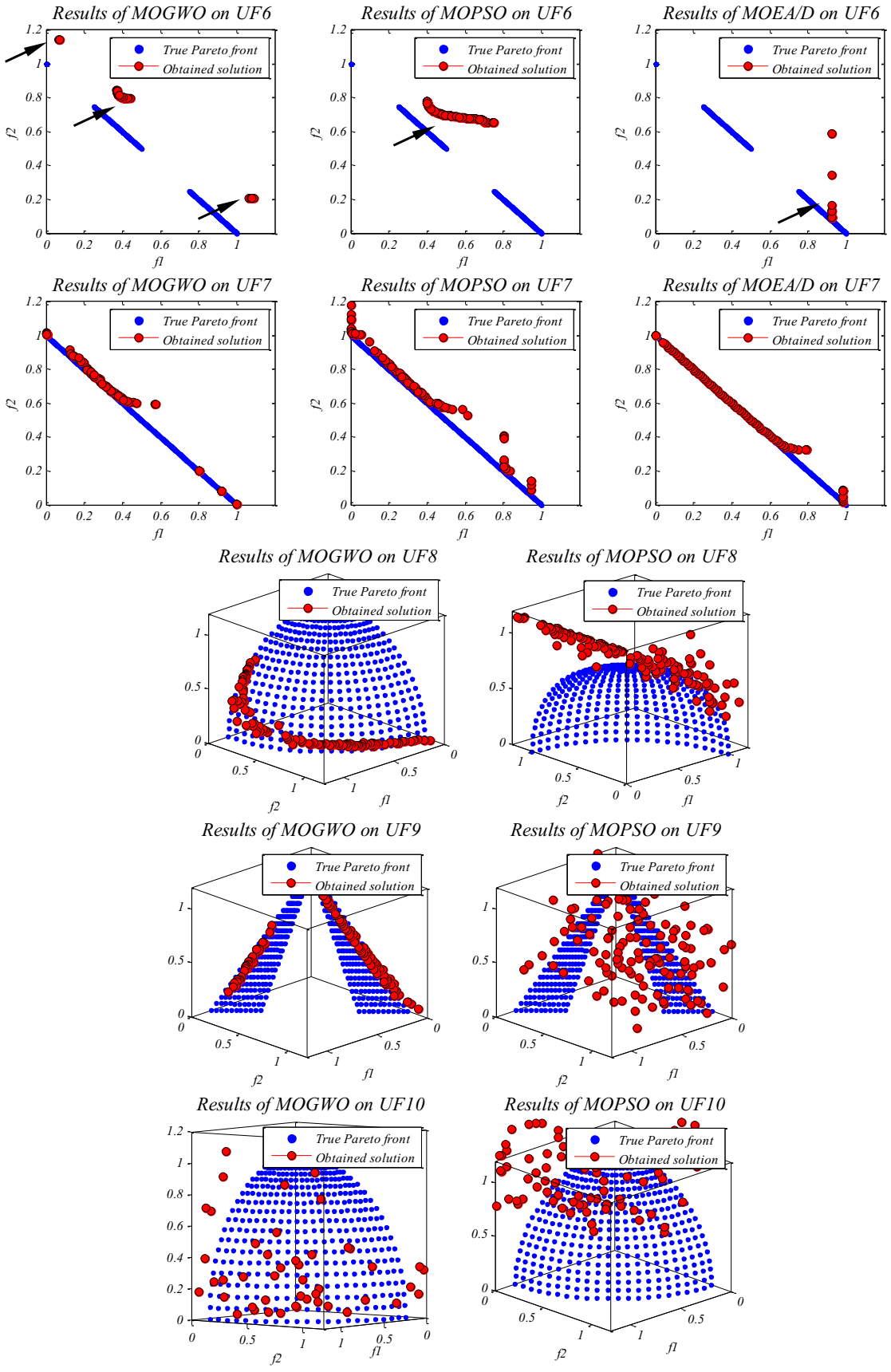

Fig. 5. Obtained Pareto optimal solutions by MOGWO, MOPSO, and MOEA/D for UF6 to UF10.

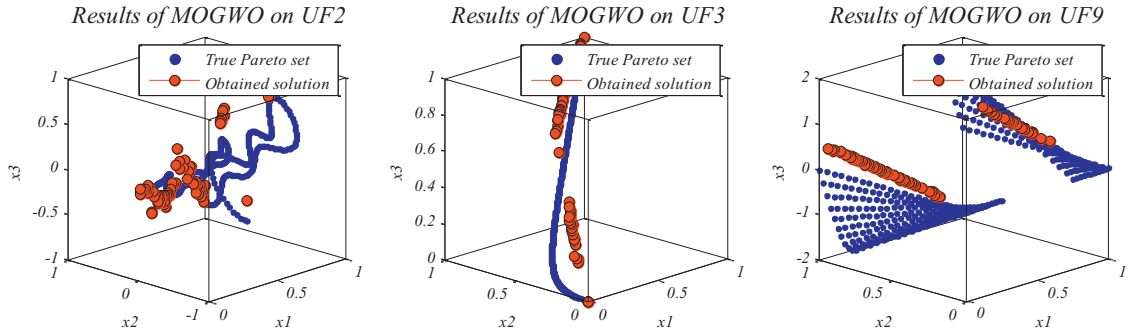

Fig. 6. True and obtained Pareto optimal sets of MOGWO on UF2, UF3, and UF9.

some experiments to see how the performance of MOGWO varies when changing this parameter. As per our observation, search agents tend to avoid each other and explore the search space proportional to the absolute value of *A*. In contrast, MOGWO searches locally and exploits the search space inversely proportional to *A*. Of course, a constant value for this parameter prevents the MOGWO from performing both exploration and exploitation. Therefore, the adaptive value is the best option unless either exploration or exploitation is beneficial. For instance, an algorithm has to always explore the search space when solving dynamic problems where the shape of search space constantly changes. Another important parameter in MOGWO is *C*. This parameter stochastically emphasizes ($C>1$) or deemphasizes ($C<1$) the effect of prey in defining the distance. This assists MOGWO to show a more random behavior throughout optimization, which causes favoring exploration and local optima avoidance. It is worth mentioning here that *C* was not linearly decreased in contrast to *A*. The *C* parameter was deliberately required to provide random values at all times in order to emphasize exploration not only during initial iterations but also final iterations. This component is very helpful in case of local optima stagnation, especially in the final iterations.

The results of MOGWO were mostly better than those of MOPSO because MOPSO only updates the position of particles with respect to one non-dominated solution obtained so far. The results prove that MOGWO was able to outperform MOPSO since it utilizes three non-dominated solutions found so far. In addition, MOPSO updates *gBest* in each iteration, so all the particles are attracted by the same (or a similar set of) *gBest(s)* in each iteration (Nebro et al., 2013). The leaders of MOGWO, however, are updated for every position update, assisting search agents to explore the search space more extensively.

The results also proved that MOEA/D algorithm mostly provides poor performance in terms of not only convergence but also coverage. However, this algorithm showed the best results on the test function with linear true Pareto optimal front. This is due to the decomposition-based nature of this algorithm that makes it highly suitable for the problems with linear fronts. This characteristic of MOEA/D yet prevents it from providing promising results on problems with convex and especially non-convex true Pareto optimal front. Since MOGWO utilizes an archive, this problem cannot be observed for this algorithm as the results evidence.

The high coverage of MOGWO was also verified in the results, which is due to the selecting leader mechanism of this algorithm. Although the selection and grid mechanism of MOGWO is similar to MOPSO, selecting three leaders for each search agent over the course of integration assists MOGWO to outperform MOPSO. This is also the reason of significant superior results of MOGWO compared to MOEA/D.

As summary, the main advantages of the proposed MOGWO algorithm compared to MOGWO and MOEA/D are high convergence and coverage. Superior convergence is due to the leader selection mechanism, in which only three non-dominated solutions always update the position of others. Another advantage is the high coverage of the MOGWO algorithm, which is because of both archive maintenance mechanism and leader selection procedure. Since the solutions are always discarded from most populated segments and leaders are chosen from the least populated segments of the archive, MOGWO improves the diversity and coverage of solutions across all objectives. Despite these benefits, MOGWO can only be applied to problems with 3 and maximum 4 objectives. Similarly to other Pareto dominance-based algorithms, MOGWO becomes less effective proportional to the number of objectives. This is due to the fact that in problems with more than 4 objectives, a large number of solutions are non-dominated, so the archive become full very quickly. Therefore, the MOGWO algorithm is suitable for solving problems with less than four objectives. In addition, this algorithm is suitable only for problems with continuous variables mainly because the GWO algorithm is for solving such problems.

5. Conclusion

This work proposed a novel multi-objective meta-heuristic called MOGWO. In fact, two new components were integrated to the GWO algorithm to allow it to perform multi-objective optimization. The first component was an archive for storing and retrieving the best non-dominated obtained solution so far during optimization. The second module was a leader selection mechanism that required MOGWO to select alpha, beta, and delta wolves for updating the position of omega wolves from the archive. The unique feature of the archive was the high emphasize on the maintenance of the solutions and updating mechanism. The grid mechanism employed improved the non-dominated solutions in the archive effectively. The leader selection mechanism proposed allowed the MOGWO algorithm to show superior coverage and convergence simultaneously. Finally, the multi-objective version of GWO was proposed utilizing the above operators for the first time in the literature. Although an archive has been utilized for other algorithms, this work integrated this non-dominated solution storage to the GWO algorithm for the first time as one of the most recent algorithms in the literature of global optimization. The archive maintenance and leader selections were also integrated to the GWO algorithm for the first time.

The proposed MOGWO algorithm was applied to 10 standard challenging CEC 2009 test problem and compared to MOPSO and MOEA/D as two well-known algorithms. The results showed that MOGWO was able to provide very competitive results. For one, the IGD performance metric quantitatively proved that MOGWO has high convergence behavior. For another, the shape of the obtained Pareto optimal solutions and coverage metrics revealed the superior coverage of the proposed algorithm qualitatively and quantitatively. According to this comprehensive study, it can be concluded that the proposed algorithm has significant merits is solving multi-objective problems, so we offer it to the researchers from different fields.

Note that the source codes of MOGWO are publicly available at http://www.alimirjalili.com/GWO.html.

For future studies, we are planning to investigate the application of the proposed algorithm in Computational Fluid Dynamics (CFD) problems. In addition, we are going to investigate the effectiveness of different robust handling methods on MOGWO that allow it to handle different types of uncertainties, which is an essential in solving real problems. Proposing a many-objective version of the MOGWO would be a valuable contribution as well.

Reference

Abbass, H. A., Sarker, R., & Newton, C. (2001). PDE: a Pareto-frontier differential evolution approach for multi-objective optimization problems. In *Proceedings of the 2001 Congress on Paper presented at the Evolutionary Computation, 2001*.

Branke, J., & Deb, K. (2005). Integrating user preferences into evolutionary multi-objective optimization. *Knowledge incorporation in evolutionary computation* (pp. 461–477). Springer.

Branke, J., Kaußler, T., & Schmeck, H. (2001). Guidance in evolutionary multi-objective optimization. *Advances in Engineering Software, 32*(6), 499–507.

Coello, C. A. C. (2009). Evolutionary multi-objective optimization: some current research trends and topics that remain to be explored. *Frontiers of Computer Science in China, 3*(1), 18–30.

Coello, C. A. C., & Lamont, G. B. (2004). Applications of multi-objective evolutionary algorithms *(*: Vol. 1. World Scientific.

Coello, C. A. C., Lamont, G. B., & Van Veldhuisen, D. A. (2007). *Evolutionary algorithms for solving multi-objective problems*. Springer.

Coello, C. A. C., Pulido, G. T., & Lechuga, M. S. (2004). Handling multiple objectives with particle swarm optimization. *Evolutionary Computation, IEEE Transactions on, 8*(3), 256–279.

Coello Coello, C. A. (2006). Evolutionary multi-objective optimization: a historical view of the field. *Computational Intelligence Magazine, IEEE, 1*(1), 28–36.

Coello Coello, C. A., & Lechuga, M. S. (2002). MOPSO: a proposal for multiple objective particle swarm optimization. In *Proceedings of the 2002 Congress on Paper presented at the Evolutionary Computation, 2002. CEC'02.*.

Das, I., & Dennis, J. E. (1998). Normal-boundary intersection: a new method for generating the Pareto surface in nonlinear multicriteria optimization problems. *SIAM Journal on Optimization, 8*(3), 631–657.

Deb, K. (2012). Advances in evolutionary multi-objective optimization. *Search Based Software Engineering* (pp. 1–26). Springer.

Deb, K., Pratap, A., Agarwal, S., & Meyarivan, T. (2002). A fast and elitist multiobjective genetic algorithm: NSGA-II. *Evolutionary Computation, IEEE Transactions on, 6*(2), 182–197.

Edgeworth, F. Y. (1881). Mathematical physics*: P.* London, England: Keagan.

Fonseca, C. M., Knowles, J. D., Thiele, L., & Zitzler, E. (2005). A tutorial on the performance assessment of stochastic multiobjective optimizers. In *Proceedings of the Third International Conference on Evolutionary Multi-Criterion Optimization (EMO 2005)*.

Gaspar-Cunha, A., & Covas, J. (2004). RPSGAe-A multiobjective genetic algorithm with elitism: application to polymer extrusion. *A Lecture Notes in Economics and Mathematical Systems volume*. Springer.

Goldberg, D. (1989). *Genetic algorithms in optimization, search and machine learning*.

Goldberg, D. E., & Holland, J. H. (1988). Genetic algorithms and machine learning. *Machine Learning, 3*(2), 95–99.

Hancer, E., Xue, B., Zhang, M., Karaboga, D., & Akay, B. (2015). A multi-objective artificial bee colony approach to feature selection using fuzzy mutual information. In *Proceedings of IEEE Congress on the Evolutionary Computation (CEC), 2015*.

Hemmatian, H., Fereidoon, A., & Assareh, E. (2014). Optimization of hybrid laminated composites using the multi-objective gravitational search algorithm (MOGSA). *Engineering Optimization, 46*(9), 1169–1182.

Kim, I. Y., & De Weck, O. (2005). Adaptive weighted-sum method for bi-objective optimization: Pareto front generation. *Structural and Multidisciplinary Optimization, 29*(2), 149–158.

Kipouros, T., Jaeggi, D. M., Dawes, W. N., Parks, G. T., Savill, A. M., & Clarkson, P. J. (2008). Biobjective design optimization for axial compressors using Tabu Search. *AIAA journal, 46*(3), 701–711.

Knowles, J., & Corne, D. (1999). The pareto archived evolution strategy: a new baseline algorithm for pareto multiobjective optimisation. In *Proceedings of the 1999 Congress on the Evolutionary Computation, 1999. CEC 99*.

Knowles, J. D., & Corne, D. W. (2000). Approximating the nondominated front using the Pareto archived evolution strategy. *Evolutionary computation, 8*(2), 149–172.

Lin, W., Yu, D., Wang, S., Zhang, C., Zhang, S., Tian, H., & . . . Liu, S. (2015). Multi-objective teaching–learning-based optimization algorithm for reducing carbon emissions and operation time in turning operations. *Engineering Optimization, 47*(7), 994–1007.

Luh, G.-C., & Chueh, C.-H. (2004). Multi-objective optimal design of truss structure with immune algorithm. *Computers & structures, 82*(11), 829–844.

Marler, R. T., & Arora, J. S. (2004). Survey of multi-objective optimization methods for engineering. *Structural and multidisciplinary optimization, 26*(6), 369–395.

Messac, A., & Mattson, C. A. (2002). Generating well-distributed sets of Pareto points for engineering design using physical programming. *Optimization and Engineering, 3*(4), 431–450.

Mirjalili, S., Mirjalili, S. M., & Lewis, A. (2014). Grey wolf optimizer. *Advances in Engineering Software, 69*, 46–61.

Nebro, A. J., Durillo, J. J., & Coello, C. (2013). Analysis of leader selection strategies in a multi-objective particle swarm optimizer. In *Proceedings of IEEE Congress on the Evolutionary Computation (CEC), 2013*.

Ngatchou, P., Zarei, A., & El-Sharkawi, M. (2005). Pareto multi objective optimization. In *Proceedings of the 13th International Conference on the Intelligent Systems Application to Power Systems, 2005*.

Osorio Velazquez, J., Coello Coello, C., & Arias-Montano, A. (2014). Multi-objective compact differential evolution. In *Proceedings of IEEE Symposium on the Differential Evolution (SDE), 2014*.

Pareto, V. (1964). *Cours d'economie politique*: Librairie Droz.

Parsopoulos, K. E., & Vrahatis, M. N. (2002). Particle swarm optimization method in multiobjective problems. In *Proceedings of the 2002 ACM symposium on Applied computing*.

Pradhan, P. M., & Panda, G. (2012). Solving multiobjective problems using cat swarm optimization. *Expert Systems with Applications, 39*(3), 2956–2964 http://dx.doi.org/10.1016/j.eswa.2011.08.157.

Rangaiah, G. P. (2008). Multi-objective optimization: techniques and applications in chemical engineering: Vol. 1. World Scientific.

Reyes-Sierra, M., & Coello, C. C. (2006). Multi-objective particle swarm optimizers: a survey of the state-of-the-art. *International Journal of Computational Intelligence Research, 2*(3), 287–308.

Schott, J. R. (1995). *Fault tolerant design using single and multicriteria genetic algorithm optimization: DTIC document*. Massachusetts Institute of Technology, Department of Aeronautics and Astronautics.

Shi, X., & Kong, D. (2015). A Multi-objective Ant Colony Optimization Algorithm Based on Elitist Selection Strategy.

Shi, Y., & Eberhart, R. (1998). A modified particle swarm optimizer. In *Proceedings of the 1998 IEEE International Conference on the Evolutionary Computation Proceedings, 1998. IEEE World Congress on Computational Intelligence*.

Sierra, M. R., & Coello, C. A. C. (2005). Improving PSO-Based multi-objective optimization using crowding, mutation andϵ-dominance. In *Proceedings of the Evolutionary Multi-Criterion Optimization*.

Srinivas, N., & Deb, K. (1994). Muiltiobjective optimization using nondominated sorting in genetic algorithms. *Evolutionary Computation, 2*(3), 221–248.

Van den Bergh, F, & Engelbrecht, A. P. (2006). A study of particle swarm optimization particle trajectories. *Information Sciences, 176*(8), 937–971.

Van Veldhuizen, D.A., & Lamont, G.B. (1998). Multiobjective evolutionary algorithm research: a history and analysis: Citeseer.

Wolpert, D. H., & Macready, W. G. (1997). No free lunch theorems for optimization. *IEEE Transactions on Evolutionary Computation, 1*(1), 67–82.

Zhang, Q., & Li, H. (2007). MOEA/D: a multiobjective evolutionary algorithm based on decomposition. *IEEE Transactions on Evolutionary Computation, 11*(6), 712–731.

Zhang, Q., Zhou, A., Zhao, S., Suganthan, P. N., Liu, W., & Tiwari, S. (2008). Multiobjective optimization test instances for the CEC 2009 special session and competition. In *Proceedings of University* of Essex, Colchester, UK and Nanyang Technological University, Singapore, Special Session on Performance Assessment of Multi-Objective Optimization Algorithms, Technical Report.

Zhou, A., Qu, B.-Y., Li, H., Zhao, S.-Z., Suganthan, P. N., & Zhang, Q. (2011). Multiobjective evolutionary algorithms: a survey of the state of the art. *Swarm and Evolutionary Computation, 1*(1), 32–49.

Zitzler, E. (1999). Evolutionary algorithms for multiobjective optimization: Methods and applications, *Vol. 63*.

Zitzler, E., Laumanns, M., & Thiele, L. (2002). SPEA2: improving the strength Pareto evolutionary algorithm for multiobjective optimization. In K. Giannakoglou et al., editors, Evolutionary Methods for Design, Optimisation and Control with Application to Industrial Problems (EUROGEN 2001). International Center for Numerical Methods in Engineering (CIMNE), pp. 95–100.

Zitzler, E., & Thiele, L. (1998). Multiobjective optimization using evolutionary algorithms—a comparative case study. In *Proceedings of the Parallel problem solving from nature—PPSN V*.

Zitzler, E., & Thiele, L. (1999). Multiobjective evolutionary algorithms: A comparative case study and the strength pareto approach. *IEEE Transactions on Evolutionary Computation, 3*(4), 257–271.