Combining Non-dominance, Objective-order and Spread Metric to Extend Firefly Algorithm to Multi-objective Optimization

M. Fernanda P. Costa[1,3], Ana Maria A.C. Rocha[2,4], and Edite M.G.P. Fernandes[4]

[1] Department of Mathematics and Applications, University of Minho, 4800-058 Guimarães, Portugal,
mfc@math.uminho.pt
[2] Department of Production and Systems, University of Minho, 4710-057 Braga, Portugal,
arocha@dps.uminho.pt
[3] Centre of Mathematics, University of Minho, 4710-057 Braga, Portugal
[4] Algoritmi Research Centre, University of Minho, 4710-057 Braga, Portugal
emgpf@dps.uminho.pt

Abstract. In this paper, we propose an extension of the firefly algorithm (FA) to multi-objective optimization. FA is a swarm intelligence optimization algorithm inspired by the flashing behavior of fireflies at night that is capable of computing global solutions to continuous optimization problems. Our proposal relies on a fitness assignment scheme that gives lower fitness values to the positions of fireflies that correspond to non-dominated points with smaller aggregation of objective function distances to the minimum values. Furthermore, FA randomness is based on the spread metric to reduce the gaps between consecutive non-dominated solutions. The obtained results from the preliminary computational experiments show that our proposal gives a dense and well distributed approximated Pareto front with a large number of points.

Keywords: Multi-objective, firefly algorithm, fitness assignment, spread metric

1 Introduction

This paper aims to extend a global optimization framework, known as firefly algorithm (FA), to tackle nonlinear multi-objective (MO) optimization problems. This is one of the most challenging problems since the goal is to optimize more than one objective. FA is a population-based algorithm and therefore suitable to solve MO problems. It is capable of finding multiple Pareto-optimal solutions in a single run. Here, we consider solving nonlinear bound constrained MO optimization problems with $no > 1$ objectives and $n \geq 1$ decision variables:

$$\begin{array}{c} \min\ (f_1(x), f_2(x), \ldots, f_{no}(x)) \\ \text{subject to}\ \ l_i \leq x_i \leq u_i,\ i = 1, \ldots, n \end{array} \tag{1}$$

where the conflicting objective functions $f_j : \mathbb{R}^n \rightarrow \mathbb{R}$, $j = 1, 2, \ldots, no$, are continuous and possibly nonlinear functions and $l \in \mathbb{R}^n$ and $u \in \mathbb{R}^n$ are the vectors of lower and upper bounds for the decision variables, respectively. We note that the feasible region $\Omega = [l, u]$ is a nonempty compact set and differentiability and convexity of the objectives are not assumed, although the search space of problem (1) is convex.

MO optimization is an important research area mainly for two reasons. First, a large number of real-world applications are formulated as MO problems; second, many issues, such as the statistical interpretation associated with performance comparison, still need to be addressed. For MO no single solution optimizes simultaneously all objectives. In practice, several conflicting objectives arise and the goal is to identify the best compromise solution among a set of Pareto-optimal solutions. The set of optimal solutions in the decision space is in general denoted as the Pareto-optimal set and its image in the objective space is denoted as Pareto-optimal front. The main task of MO algorithms is to support a decision maker to formulate his/her preferences and to identify the best of the Pareto-optimal solutions.

In a MO minimization problem, a solution $\bar{x} \in \mathbb{R}^n$ is said to dominate $\hat{x} \in \mathbb{R}^n$ if and only if $f_j(\bar{x}) \leq f_j(\hat{x})$ for all $j \in \{1, 2, \ldots, no\}$ where $f_j(\bar{x}) < f_j(\hat{x})$ for at least one j. Further, a solution $\bar{x} \in \mathbb{R}^n$ is said to be Pareto-optimal if and only if there is no solution $\hat{x} \in \mathbb{R}^n$ that dominates $\bar{x}$. Thus, the goal with a MO algorithm is to find a good and balanced approximation to the set of Pareto-optimal solutions.

The most popular methods to tackle a MO problem are based on the aggregation of the objectives, on the ϵ–constraint, and on producing an approximation to the Pareto-optimal front directly. The aggregation method transforms the MO formulation into a uni-objective formulation problem by assigning to each objective function f_j a non-negative weight w_j such that $\sum_{j=1}^{no} w_i = 1$, and minimizing an aggregate function that is the weighted sum of the objectives. The approximate Pareto-optimal front is obtained by running as many times as the desired number of points using different weight values [22]. In the ϵ–constraint method one objective is selected to be minimized and all the other objective functions are converted into inequality constraints by setting an upper value to each one [13]. Methods to compute an approximation to the Pareto front in a single run are in general stochastic population-based search techniques. Fitness assignment is a crucial issue in MO algorithms and depends on the entire set of points in the population. Two categories of common strategies to assign fitness are aggregation-based and Pareto/dominance-based. The latter may use more than one dominance order (for example, dominance rank, dominance count or dominance depth) [35]. Fitness assignment strategies may also depend on MO performance metrics, for instance, the hyper-volume, the purity metric or the spread metric [13,16,33,34].

Evolutionary algorithms are widely used when solving MO optimization problems. They are designated as MO evolutionary algorithms (MOEAs) and largely dominate the research area of approximate metaheuristics for MO [16]. From the most classical procedure VEGA to other more recent MOEA variants, like MOGA, MOMGA, NPGA, NSGA, PESA, PAES, SPEA, NSGA-II, SPEA2, RPSGAe and MEGA [8,12,17,33,36], all of them have been used in a variety of real-world applications. In [23], a hybrid multi-objective evolutionary algorithm combining a genetic algorithm and a particle swarm optimization is presented; in [4,11], different robust MO optimization procedures are presented; and in [14], robustness assessment during multi-objective optimization using a MOEA is discussed. Besides MOEAs other metaheuristics have been used in MO optimization [1,6,7,22]. Deterministic-type approaches are also available [5,10].

The contribution of this paper is the extension of the FA paradigm to the MO optimization. FA is a recently developed bio-inspired metaheuristic algorithm that is capable of computing global solutions to optimization problems [15,27,28]. It is a swarm intelligence optimization algorithm inspired by the flashing behavior of fireflies at night, and it competes with the most well-known swarm intelligence algorithms, like ant colony optimization, particle swarm optimization, artificial bee colony, artificial fish swarm, bat algorithm and cuckoo-search.

FA has already been adapted to the MO optimization area [2,20,29]. Recently proposed FA extensions to MO are related with applications in operations research, like fleet planning problems, circuit design problems, production scheduling system, economic emission dispatch problem [31], energy optimization in grid environments [3], hybrid flowshop scheduling problem [21], job shop scheduling problems [18], geometric design of clamped-free beams [19] and optimal hydrocyclone design [25]. Most of these studies transform the MO formulation into a uni-objective one, although others produce approximations to the Pareto front in a single run using an aggregation-based strategy to assign fitness to points.

Our proposal for the MO optimization area uses a non-dominance/dominance ranking combined with an objective-order process based on scaled distances to the minimum values for the fitness assignment procedure. It also incorporates a spread metric-based randomness term into the FA paradigm to generate candidate points from the current ones. This randomness term aims to diversify the search as well as to reduce the gaps between consecutive non-dominated solutions in the approximated Pareto front. The herein proposed non-dominance/dominance ranking aims to favor non-dominated points of the populations giving them ranks that are always lower than those assigned to any of the other dominated points. This way, non-dominated points correspond to the positions of the brightest fireflies. Our algorithm computes candidate points to all current ones, except to the best point of the population, representing the position of the brightest firefly of all. Assuming that all non-dominated and dominated positions in the search space are ordered, the algorithm simulates movements to all fireflies, except the brightest, in direction to the more brighter ones. Then each computed candidate/trial position is accepted just after the movement except when a current non-dominated position generates a dominated trial position. Furthermore, at the end of each iteration, the set of non-dominated solutions found thus far is updated with the accepted non-dominated points, being the dominated solutions removed from the set. Our proposal is designated by Multi-Objective-order Firefly Algorithm (MOoFA).

The remaining part of the paper is organized as follows. In Section 2 the FA paradigm is described and in Section 3 the proposed MOoFA is presented and discussed. Section 4 reports on the preliminary computational experiments carried out using a benchmark set of MO problems and we conclude the paper with Section 5.

2 The FA paradigm

Throughout the paper, $\|\cdot\|$ represents the Euclidean norm of a vector and the vector $x = (x_1, x_2, \ldots, x_n)^T$ represents the position of a firefly in the search space. The position of the firefly j will be represented by $x^j \in \mathbb{R}^n$. We assume that the size of the population of fireflies is $1 < m < \infty$. In the context of an uni-objective optimization problem, firefly j is brighter than firefly i if the objective function value at x^j is lower than the objective value at x^i.

FA is a bio-inspired metaheuristic algorithm inspired by the flashing behavior of fireflies at night. According to [9,26,27,28,30,32], the three main rules used to construct the standard algorithm are the following: (i) all fireflies are unisex, meaning that any firefly can be attracted to any other brighter one; (ii) the brightness of a firefly is determined from the encoded objective function; (iii) attractiveness is directly proportional to brightness but decreases with distance. In the FA paradigm, the movement of a firefly i is attracted to another more attractive/brighter firefly j and the new candidate position, also designated by trial position, for firefly i is given by:

$$t^i = x^i + \beta(x^j - x^i) + \alpha\left(z + L(0,1)\sigma^i\right), \tag{2}$$

where x^i represents its current position, $\alpha \in [0, 1]$ and

$$\beta = \beta_0 \exp\left(-\gamma \|x^i - x^j\|^2\right) \tag{3}$$

is the attractiveness of a firefly which varies with the light intensity seen by adjacent fireflies and the distance between themselves. The parameter β_0 is the attraction parameter when the distance is zero. $L(0, 1)$ is a random number from the standard Lévy distribution centered at zero with an unitary standard deviation. The vector $z = z(k)$ is a reference point from the set of best solutions found so far and the vector $\sigma^i = \left(|x^i_1 - z_1|, \ldots, |x^i_n - z_n|\right)^T$ gives the variation around z. The notation $z(k)$ means that it varies with the iteration counter, k, of the algorithm. The second term on the right hand side of (2) is due to the attraction while the third term gives randomness, with α being a scale parameter that controls the randomness and aims to maintain the diversity of solutions. The parameter γ characterizes the variation of the attractiveness, and is crucial to speed the convergence of the algorithm. As in [9], we allow α to decrease linearly with k, from $\alpha_{\max}$ to $\alpha_{\min}$, and we use a dynamic update of γ that increases the attractiveness with k from a lower value $\gamma_{\min}$ to an upper value $\gamma_{\max}$. Contrary to the evolutionary strategies and genetic algorithm, in FA all firefies simulate movement in order to find a better position. Although in the oldest versions of FA, the brightest firefly was not moved, some recent versions move it, either randomly or in a direction in which the brightness increases [26,30]. Furthermore, the new positions of each firefly are only accepted if they improve over the old ones. This is particularly promising since the best position is never lost.

3 Strategies in MOoFA

Since the proposed MOoFA is of a stochastic nature, the goal is to search for the best approximation to the Pareto-optimal front. MOoFA performs the search in the objective space, i.e., the algorithm selects the positions to be varied (corresponding to fireflies that simulate movement) based on the fitness assigned to the fireflies in the population. This fitness assignment is a crucial issue in FA since a firefly movement depends on brighter fireflies and the brightness is inversely proportional to the fitness value. In this extended FA for MO, the lower the fitness value the brighter is the firefly (and the lower is the order of the position). The simplest way to implement FA in a MO paradigm is to order the positions of fireflies from lowest to highest fitness value. In this paper, we propose two fitness assignment schemes that are based on an ordering strategy of the objective values. To order the positions of the fireflies, the following ranking steps are required.

1. Assign 'non-dominance rank', r_{n-d}, that aims to favor non-dominated points giving them the rank value $r_{n-d} = 1$, and giving to the remaining (the dominated ones) points $r_{n-d} = 2$;
2. Assign 'f–values order', o_f, that aims to give lower order to points with lower function values. Two schemes are proposed. One depends on assigning ranks to the objective function values; the other relies on the difference from the function values themselves to the minimum value. The 'f–values order' aggregates quantities using weights that satisfy $0 \leq w_j \leq 1$ and $\sum_{j=1}^{no} w_j = 1$. Thus,
   (a) Using ranks (integer values ranging from 1 to m), r_j, assigned to the objective values $f_j(x^i)$, $j = 1, \ldots, no$, the 'f–values order' of a point x^i is calculated by

$$o_f(x^i) = \frac{1}{m} \left(w_1 \mathrm{r}_1 + w_2 \mathrm{r}_2 + \ldots + w_{no} \mathrm{r}_{no} \right). \tag{4}$$

   (b) Using the objective function values, a factor that is a scaled distance to the minimum value of objective f_j is computed,

$$\mathrm{s}_j = \frac{f_j(x^i) - f_{j,\min}}{f_{j,\max} - f_{j,\min}} \tag{5}$$

   where $0 \leq \mathrm{s}_j \leq 1$, $f_{j,\max}$ and $f_{j,\min}$ are the maximum and minimum values of f_j attained by the population, respectively. Then, the 'f–values order' is computed by

$$o_f(x^i) = w_1\mathrm{s}_1 + w_2\mathrm{s}_2 + \ldots + w_{no}\mathrm{s}_{no}. \tag{6}$$

3. Finally, for either case (2a) or (2b), the fitness value, $Fit(x^i)$, assigned to each point x^i is defined by

$$Fit(x^i) = r_{n-d} + o_f(x^i). \tag{7}$$

This way non-dominated points have fitness values in the range $[1, 2]$ and dominated points have fitness in the range $[2, 3]$.

Table 1 shows the fitness assignment scheme (4), for a small example with two objectives, ten points in the population, and $w_1 = w_2 = 0.5$. The last column in the table shows the ordering of the points based on the Fit values. (We note that any occurring tie is broken arbitrarily.) Table 2 depicts the fitness assignment scheme based on (5) and (6). We note that this ordering is not the same as that of previous table. In Table 1 two sets of ties occur in Fit: one originates x^5 and x^6, the other x^8 and x^9. With the factor s_j, the likelihood that ties will occur is much lower than with the scheme (4).

Table 1: Fitness assignment based on ranking the objectives (4), for ten points.

i	$f_1(x^i)$	$f_2(x^i)$	r_1	r_2	$o_f(x^i)$	r_{n-d}	$Fit(x^i)$	ordering
1	6.75	3	6	7	0.65	2	2.65	x^5
2	**4**	**1**	1	3	0.20	1	1.20	x^1
3	**7**	**0.5**	7	1	0.40	1	1.40	x^3
4	10	2.5	10	5	0.75	2	2.75	x^9
5	5	4	3	10	0.65	2	2.65	x^6
6	4.5	2	2	4	0.30	2	2.30	x^4
7	**6**	**0.75**	4	2	0.30	1	1.30	x^2
8	6.5	3.5	5	9	0.70	2	2.70	x^7
9	9	2.7	9	6	0.75	2	2.75	x^8
10	8	3.25	8	8	0.80	2	2.80	x^{10}

Non-dominated points are in bold style

We now briefly describe some technical issues of MOoFA in Algorithm 1. MOoFA starts by randomly generating m points – positions of the population of fireflies – in the search space Ω. The objective functions are evaluated at all points and the non-dominated points are identified. The set, denoted by ND, of all produced non-dominated points (the corresponding no–tuple $(f_1, f_2, \ldots, f_{no})$) is initialized. The fitness assignment strategy described in (7) is applied and the points are ordered according to their fitness value Fit, from lowest to largest, i.e., x^1 is the point with lowest Fit value, x^2 is the point with the second lowest value of Fit, and so forth, x^m is the point with largest Fit value. Now, new candidate positions are computed for the current position x^2 and all the others that follow, i.e., x^2 may be moved towards x^1 (meaning that firefly 2 is attracted

Table 2: Fitness assignment based on scaled distance of objectives to minimum (5) and (6), for ten points.

i	$f_1(x^i)$	$f_2(x^i)$	s_1	s_2	$o_f(x^i)$	r_{n-d}	$Fit(x^i)$	ordering
1	6.75	3	0.4583	0.7143	0.5863	2	2.5863	x^6
2	**4**	**1**	0	0.1429	0.0714	1	1.0714	x^1
3	**7**	**0.5**	0.5	0	0.2500	1	1.2500	x^3
4	10	2.5	1	0.5714	0.7857	2	2.7857	x^{10}
5	5	4	0.1667	1	0.5833	2	2.5833	x^5
6	4.5	2	0.0833	0.4286	0.2560	2	2.2560	x^4
7	**6**	**0.75**	0.3333	0.0714	0.2024	1	1.2024	x^2
8	6.5	3.5	0.4167	0.8571	0.6369	2	2.6369	x^7
9	9	2.7	0.8333	0.6286	0.7310	2	2.7310	x^9
10	8	3.25	0.6667	0.7857	0.7262	2	2.7262	x^8

Non-dominated points are in bold style

to firefly 1), x^3 may be moved towards x^1 (in first place) and then x^2, and so on. We use the term 'candidate' because, in the proposed FA extension to MO, the new point may not be a promising position, when compared with the current one, and will not be accepted. This is a crucial issue and arises when a non-dominated current point generates a dominated candidate. In all the other cases, the candidate position is accepted. Furthermore, whenever a position is declared non-dominated, via flag='true' in the algorithm, any subsequent candidate position will be accepted only if it is non-dominated.

When extending FA to MO, the choice of the point z to center the randomization contribution to the firefly movement (see equation (2)) is based on a MO performance measure, a spread metric. Thus, z is one of the arguments of two consecutive non-dominated points with a maximum distance (based on infinity norm) in objective function values, i.e.,

$$z = \arg \max_{j \in \{1,\ldots,no\}} \left\{ \max_{i \in \{1,\ldots,|ND|-1\}} \left\{ f_j^{i+1} - f_j^i \right\} \right\} \tag{8}$$

where $|ND|$ is the cardinal of ND. Our choice falls on the first of the two points. We recall that the f_j values are sorted (from lowest to largest). This choice for the point z aims to force the movement of the firefly i towards the set of non-dominated points, as well as to the region where the distance between consecutive points is largest. This way the algorithm will generate an approximated Pareto front with evenly spread points. Only after all points (except x^1) have potentially moved towards other points, is the set ND updated with the new accepted non-dominated points, being removed the dominated solutions. Finally, at the end of each iteration, all accepted points are ordered based on their fitness values.

The output of the algorithm is the set ND that contains an approximation to the Pareto front.

```
Data: k_max, m
Set k = 1;
Randomly generate x^i ∈ Ω, i = 1,...,m and evaluate f_j(x^i), i = 1,...,m,
j = 1,...,m;
Define the set ND with the non-dominated points;
Assign flag='true' to all non-dominated points of the population;
Assign fitness to all m points, using (7), and order them;
while k ≤ k_max do
    forall x^i  such that  i = 2,...,m do
        forall x^j  such that  j = 1,...,i-1 do
            Compute randomization term and attractiveness β;
            Move firefly i towards j using (2) and evaluate
            f_j(t^i), j = 1,...,no;
            if x^i has flag='true' then
                if t^i is a non-dominated point then
                    Set x^i = t^i and assign flag='true' to x^i;
                end
            else
                Set x^i = t^i;
                if x^i is a non-dominated point then
                    Assign flag='true' to x^i;
                end
            end
        end
    end
    Set k = k + 1;
    Update the set ND with the accepted non-dominated points (remove
    the dominated ones);
    Assign flag='true' to all non-dominated points of the population;
    Assign fitness to all m points, using (7), and order them;
end
Output: set ND
```

Algorithm 1: MOoFA

The algorithm MOoFA stops when a target number of iterations, $k_{\max}$, is exceeded, although other criteria may be used. We may require that the number of function evaluations reaches a target value, or the largest gap between two consecutive points of the approximated Pareto front falls below a tolerance.

4 Numerical Comparisons

MOoFA is coded in MATLAB programming language (Matlab Version 8.1.0.604 (R2013a)) and the numerical experiments were carried out on a PC Intel Core

2 Duo Processor E7500 with 2.9GHz and 4Gb of memory. To analyze the performance of two variants of MOoFA, a set of nine benchmark problems with different properties in terms of Pareto-optimal front is used (see [12,29,34]). The known acronyms are: FON with non-convex Pareto front, $n = 3$ and $no = 2$; KUR with discontinuous Pareto front, $n = 3$ and $no = 2$; POL with discontinuous Pareto front, $n = 2$ and $no = 2$; SCH with convex Pareto front, $n = 1$ and $no = 2$; ZDT1 with convex Pareto front, $n = 30$ and $no = 2$; ZDT2 with non-convex Pareto front, $n = 30$ and $no = 2$; ZDT3 with discontinuous Pareto front, $n = 30$ and $no = 2$; ZDT4 with convex Pareto front, $n = 10$ and $no = 2$; ZDT6 with non-convex Pareto front and $n = 10$ and $no = 2$. We use the following acronyms to identify the two variants of MOoFA: (i) 'MOoFA-rank', for Algorithm 1 based on the objective ranking (4), with fitness (7); (ii) 'MOoFA', for Algorithm 1 based on the scaled objective distance to the minimum (5) and (6), with fitness (7). Each tested variant was run 10 times with each problem. In Algorithm 1, we set $m = 50$, as suggested in [29], and $k_{\max} = 100$ when solving FON, KUR, POL, SCH, ZDT1, ZDT2, ZDT3 and ZDT6, and $m = 100$ and $k_{\max} = 500$ when solving ZDT4. Some preliminary experiments were carried out to analyze the performance of the algorithms using previously proposed parameter values [9,15]. The results showed that higher quality solutions are obtained with $\beta_0 = 1$, $\alpha_{\min} = 0.01$, $\alpha_{\max} = 0.5$, $\gamma_{\min} = 0.1$ and $\gamma_{\max} = 10$ as presented in [9].

4.1 MO performance measures

Three aspects could be considered when comparing the performance of multi-objective optimization algorithms: (i) the closeness to the true Pareto front; (If the true Pareto front for a given problem is known then the closeness can be measured using, for instance, the distance between the true Pareto front and the produced approximation to the Pareto front.) (ii) the spread along the Pareto front; (iii) the number of solutions in the non-dominated set. Here, we aim to compare closeness to the true Pareto front and select two performance metrics known as generational distance, GD_p, and inverted generational distance, IGD, which are defined by

$$\mathrm{GD}_p = \frac{1}{|ND|}\left(\sum_{j=1}^{|ND|} d_j^p\right)^{1/p} \quad \text{and} \quad \mathrm{IGD} = \frac{1}{N}\left(\sum_{j=1}^{N} D_j\right) \tag{9}$$

respectively, where $p \geq 1$, d_j is the Euclidean distance from the j-th point of the approximated front ND to its nearest point of the true Pareto front [12,13,22,29], D_j is the minimum Euclidean distance between the point j in the true Pareto front and the points in ND and N is the number of uniformly distributed points along the true Pareto front. Smaller values of GD_p and IGD indicate better approximations to the Pareto-optimal front.

4.2 Experimental results

First, using a visual presentation of our results, we show the approximated Pareto front produced by 'MOoFA-rank' and 'MOoFA'. We plot the ND set that corresponds to the run that gave the lowest GD_2 (corresponding to $p = 2$) value. Figure 1 contains the six plots that are produced by Algorithm 1 and objectives ranking (4), when solving SCH, ZDT1, ZDT2, ZDT3, ZDT4 and ZDT6.

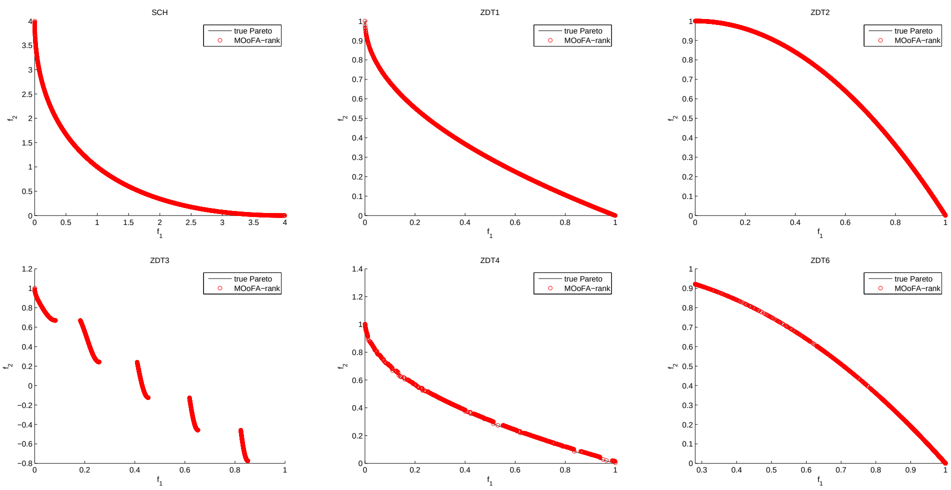

Fig. 1: Approximated Pareto front produced by Algorithm 1 and objective ranking (4).

Figure 2 contains the plots for the six previously referred problems using Algorithm 1 and objective distances to the minimum values (5) and (6). We may conclude that the produced approximated Pareto fronts are dense and have a sufficient large number of uniformly distributed points. The differences between the two variants are not significant, although we observe a slight improvement on closeness and density of MOoFA front, for the problems ZDT4 and ZDT6.

The large number of non-dominated solutions produced by Algorithm 1 requires a moderate computational effort specially when $m = 100$ and the algorithm runs for 500 iterations. We then decided to test another variant that computes candidate solutions only to fireflies that correspond to dominated positions. This means that only the dominated fireflies are attracted to non-dominated and dominated brighter fireflies. Hence, if $m_{nd} \leq m$ represents the number of non-dominated positions in the current population, the outer 'for' loop in Algorithm 1 starts with $x^{m_{nd}+1}$ and finishes with x^m. This variant is denoted by 'MOoFA-dom'. We observed that this variant produced a very small number of non-dominated solutions. However, increasing the size of the population and the maximum number of iterations allow the variant to find a larger number of points while improving GD_p and IGD. Thus, we have used $m = 100$ and $k_{\max} = 500$ for all tested problems. Figure 3 displays the plots that correspond to the previously referred six problems. Nevertheless, these results are not as good as those produced by the variants 'MOoFA-rank' and 'MOoFA' of Algorithm 1.

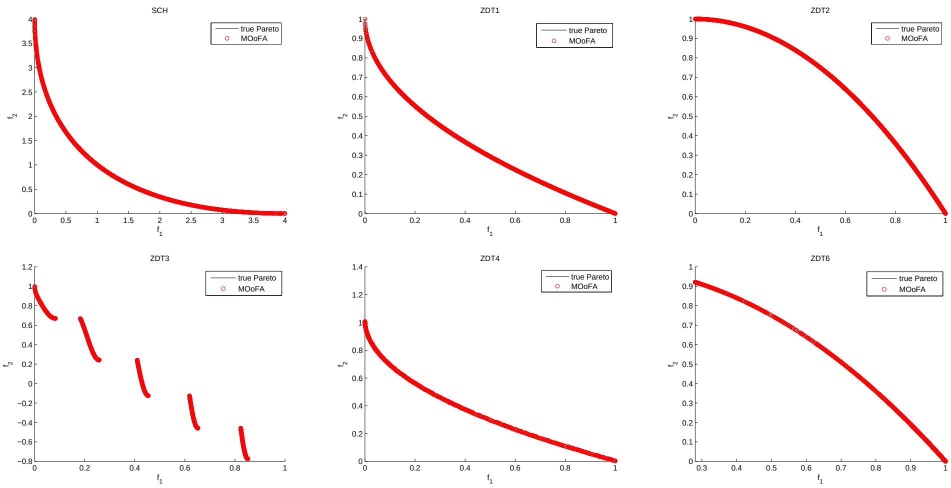

Fig. 2: Approximated Pareto front produced by Algorithm 1 and objective distance to the minimum (5) and (6).

Now, we report on Tables 3 and 4 the numerical results produced by 'MOoFA-rank' and 'MOoFA'. For these comparisons we use both the generational distance GD_2 (based on $p = 2$), GD_1 (based on $p = 1$) and the inverted generational distance IGD (see (9)).

Table 3 contains the corresponding averaged GD_2 values over the runs. In parentheses, we show the average number of non-dominated solutions $|ND|$. The other results for comparison are from MOFA and three popular MOEAs known as SPEA, NSGA-II and DEMO, that are available from [29]. The author in [29] reports the use of $m = 50$, $k_{\max} = 500$, and in FA several values for α_0 (ranging from 0.1 to 0.5) and β_0 (ranging from 0.7 to 1) were tested, with $\alpha = \alpha_0(0.9)^k$. Our results (based on $N = 500$) show that the variant 'MOoFA' gives slightly better values of GD_2 on problems ZDT1, ZDT2, ZDT3 and ZDT6 and variant 'MOoFA-rank' is better on SCH and ZDT4. Furthermore, when compared with MOFA [29], SPEA, NSGA-II and DEMO, our proposed variants of MOoFA give lower averaged GD_2 values when solving problems ZDT1, ZDT2 and ZDT3, but larger values when solving SCH and compared with MOFA and DEMO.

Table 4 contains average values of GD_1 and IGD computed from our results. We now compare with the GD_1 results reported in [12] for SPEA, PAES and NSGA-II, where $m = 100$, $k_{\max} = 250$ are used. The results obtained with the

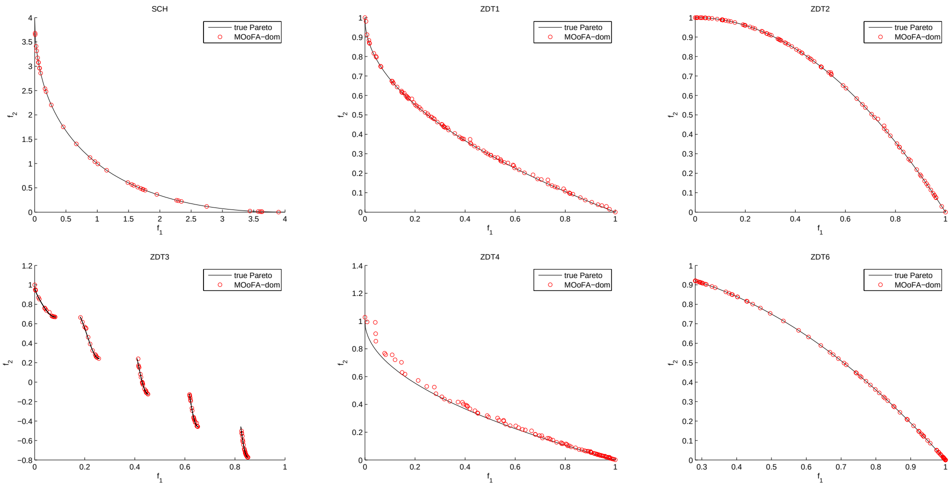

Fig. 3: Approximated Pareto front produced by 'MOoFA-dom', and objective distance to the minimum (5) and (6).

problems FON, KUR and POL are also shown for comparison. We remark that the reference Pareto fronts of problems FON, KUR and POL were obtained from the literature and they are not uniformly distributed. We also note that the IGD values produced by our variants of the Algorithm 1, when solving POL, are large since the set ND has just a few points with $f_1 > 15$ and $f_2 < 0.1$. When comparing GD_1, NSGA-II has slightly lower values on problems FON, KUR and ZDT4, PAES has a lower value on SCH, while the variant 'MOoFA' produces lower values than any of the other four in comparison, when solving problems POL, ZDT1, ZDT2, ZDT3 and ZDT6.

Our final conclusions are that MOoFA (based on Algorithm 1) is able to produce competitive results and provides dense and well distributed approximated Pareto front with a large number of points.

Table 3: Comparison based on GD_2 with $|ND|$ in parentheses.

	'MOoFA-rank'	'MOoFA'	MOFA†	SPEA†	NSGA-II†	DEMO†
Prob.	GD_2	GD_2	GD_2	GD_2	GD_2	GD_2
SCH	2.37e-04 (3314)	2.57e-04 (2977)	4.55e-06	5.17e-03	5.73e-03	1.79e-04
ZDT1	3.35e-05 (2644)	2.09e-05 (3325)	1.90e-04	1.78e-03	3.33e-02	1.08e-03
ZDT2	1.96e-05 (3360)	1.35e-05 (3517)	1.52e-04	1.34e-03	7.24e-02	7.55e-04
ZDT3	2.12e-05 (3110)	1.98e-05 (2639)	1.97e-04	4.75e-02	1.14e-01	1.18e-03
ZDT4	3.63e-01 (1201)	6.59e-01 (1033)	–	–	–	–
ZDT6	4.68e-03 (2033)	1.59e-04 (4402)	–	–	–	–

† results available in [29] with $m = 50$ and $k_{\max} = 500$; – not available

Table 4: Comparison based on GD_1 and IGD.

	'MOoFA-rank'		'MOoFA'		SPEA‡	PAES‡	NSGA-II‡
Prob.	GD_1	IGD	GD_1	IGD	GD_1	GD_1	GD_1
FON	9.50e-03	3.57e-03	8.57e-03	3.47e-03	1.26e-01	1.51e-01	1.93e-03
KUR	3.37e-02	4.15e-02	3.51e-02	3.78e-02	4.56e-02	5.73e-02	2.90e-02
POL	1.13e-02	1.57e+04	1.12e-02	1.57e+04	3.78e-02	3.09e-02	1.56e-02
SCH	5.05e-03	1.10e-03	5.40e-03	1.24e-03	3.40e-03	1.31e-03	3.39e-03
ZDT1	1.11e-03	5.29e-04	8.42e-04	3.57e-04	1.80e-03	8.21e-02	3.35e-02
ZDT2	9.63e-04	4.09e-04	7.11e-04	6.12e-02	1.34e-03	1.26e-01	7.24e-02
ZDT3	8.95e-04	3.13e-04	8.22e-04	1.06e-01	4.75e-02	2.39e-02	1.15e-01
ZDT4	3.91e+00	2.27e+00	1.02e+01	8.23e+00	7.34e+00	8.55e-01	5.13e-01
ZDT6	1.13e-02	5.16e-03	9.03e-04	5.81e-04	2.21e-01	8.55e-02	2.97e-01

‡ results available in [12] with $m = 100$ and $k_{\max} = 250$

5 Conclusions

We have presented a new methodology to solve nonlinear bound constrained MO optimization problems based on the FA paradigm, on non-dominance/dominance ranking and aggregation of objective function distances to the minimum values, for fitness assignment, and on the spread metric to reduce the gaps between consecutive non-dominated solutions. MO benchmark problems of the literature were selected to test our proposal. From the obtained results we have found out that the algorithm is effective and worthy of further research. The obtained values for the generational distance to the true Pareto front were rather competitive although distance alone is not sufficient for performance assessment. Thus, this study will be complemented with other performance guided metrics.

Future work will focus on incorporating a clustering technique into MOoFA to reduce the number of archived non-dominated solutions while maintaining the good density-based characteristics, so that computational time can be reduced. Furthermore, experimental tests will be extended to MO problems with three and more objectives and larger number of variables. The effect of increasing the number of objectives on the convergence of the algorithm will be investigated. Results available in the literature from other MOEAs will be used for comparison purposes.

Acknowledgments. The authors thank the anonymous referees for the valuable suggestions. This work has been supported by FCT (Fundação para a Ciência e Tecnologia, Portugal) in the scope of the projects: PEst-OE/MAT/UI0013/2014 and PEst-OE/EEI/UI0319/2014.

References

1. Akbari, R.B., Hedayatzadeh, R., Ziarati, K., Hassanizadeh, B.: A multi-objective artificial bee colony algorithm. Swarm and Evolutionary Computation 2, 39–52 (2012)
2. Amiri, B., Hossain, L., Crawford, J.W., Wigand, R.T.: Community detection in complex networks: multi-objective enhanced firefly algorithm. Knowl.-Based Syst. 46, 1–11 (2013)
3. Arsuaga-Ríos, M., Vega-Rodríguez, M.A.: Multi-objective firefly algorithm for energy optimization in grid environments. In Swarm Intelligence, Dorigo, M. et al (eds.) Lect. Notes Comput. Sc. vol. 7461, 350–351 (2012)
4. Barrico, C., Antunes, C.H.: Robustness analysis in multi-objective optimization using a degree of robustness concept. Proceedings of the 2006 IEEE World Congress on Computational Intelligence (WCCI 2006), 6778–6783 (2006)
5. Biondi, T., Ciccazzo, A., Cutello, V., D'Antona, S., Nicosia, G., Spinella, S.: Multi-objective evolutionary algorithms and pattern search methods for circuit design problems. J. Univers. Comput. Sci. 12(4), 432–449 (2006).
6. Chica, M., Cordón, O., Damas, S., Bautista, J.: A new diversity induction mechanism for a multi-objective ant colony algorithm to solve a real-world time and space assembly line balancing problem. Memetic Comp. 3, 15–24 (2011)
7. Coello, C.A.C., Pulido, G.T., Lechuga, M.S.: Handling multiple objectives with particle swarm optimization. IEEE T. Evolut. Comput. 8(3), 256–279 (2004)
8. Costa, L., Oliveira, P.: An elitist genetic algorithm for multiobjective optimization. In Metaheuristics, Resende, M.G.C., Pinho de Sousa, J. (eds.), Kluwer Academic Publishers, USA, 217–236 (2004)
9. Costa, M.F.P., Rocha, A.M.A.C., Francisco, R.B., Fernandes, E.M.G.P.: Heuristic-based firefly algorithm for bound constrained nonlinear binary optimization. Advances in Operations Research 2014, Article ID 215182, 12 pages (2014)
10. Custódio, A.L., Madeira, J.F.A., Vaz, A.I.F., Vicente, L.N.: Direct multisearch for multiobjective optimization. SIAM J. Optim. 21, 1109–1140 (2011)
11. Deb, K., Gupta, H.: Introducing robustness in multi-objective optimization. Evolut. Comput. 14(4), 463–494 (2006)
12. Deb, K., Pratap, A., Agrawal, S., Meyarivan, T.: A fast and elitist multiobjective genetic algorithm: NSGA-II. IEEE T. Evolut. Comput. 6(2), 182–198 (2002)
13. Denysiuk, R.: Multiobjective Optimization: Review, Algorithms, and Applications. Ph.D. Thesis, University of Minho, Braga, Portugal (2014)
14. Ferreira, J., Fonseca, C.M., Covas, J.A., Gaspar-Cunha, A.: Evolutionary multi-objective robust optimization. In Advances in Evolutionary Algorithms, Kosiński, W. (ed.), 261–278, I-Tech Education and Publishing (2008)
15. Fister, I., Fister Jr., I., Yang, X.-S., Brest, J.: A comprehensive review of firefly algorithms. Swarm and Evolutionary Computation 13, 34–46 (2013)
16. Fontes, D.B.M.M., Gaspar-Cunha, A.: On multi-objective evolutionary algorithms. In Handbook of Multicriteria Analysis, Zopounidis, C., Pardalos, P.M. (eds.) Appl. Optimizat. vol. 103, 287–310 Springer-Verlag (2010)
17. Gaspar-Cunha, A, Covas, J.A.: RPSGAe – reduced Pareto set genetic algorithm: application to polymer extrusion. In Metaheuristics for Multiobjective Optimisation, Gandibleux, X., Sevaux, M., Sörensen, K., T'kindt, V. (eds.), Lect. Notes Econ. Math. vol. 535, 221–249 (2004)
18. Karthikeyan, S., Asokan, P., Nickolas, S., Page, T.: A hybrid discrete firefly algorithm for solving multi-objective flexible job shop scheduling problems. International Journal of Bio-Inspired Computation (in press) (2014)

19. Lobato, F.S., Arruda, E.B., Cavalini Jr., A.Ap., Steffen Jr., V.: Engineering system design using firefly algorithm and multi-objective optimization. In ASME 2011, vol. 2: 31th Computers and Information in Engineering Conference (Parts A and B) 577–585 (2011)
20. Li, H., Ye, C.: Firefly algorithm on multi-objective optimization of production scheduling system. Advances in Mechanical Engineering and its Applications 3(1), 258–262 (2012)
21. Marichelvam, M.K., Prabaharan, T., Yang, X.-S.: A discrete firefly algorithm for the multi-objective hybrid flowshop scheduling problems. IEEE T. Evolut. Comput. 18(2), 301–305 (2014)
22. Molina, J., Laguna, M., Martí, R., Caballero, R.: SSPMO: a scatter tabu search procedure for non-linear multiobjective optimization. INFORMS J. Comput. 19(1), 91–100 (2007)
23. Mousa, A.A., El-Shorbagy, M.A., Abd-El-Wahed, W.F.: Local search based hybrid particle swarm optimization algorithm for multiobjective optimization, Swarm and Evolutionary Computation 3, 1–14 (2012)
24. Schutze, O., Esquivel, X., Lara, A., Coello Coello, C.A.: Using the averaged Hausdorff distance as a performance measure in evolutionary multiobjective optimization. IEEE T. Evolut. Comput. 16(4), 504–522 (2012)
25. Silva, D.O., Vieira, L.G.M., Lobato, F.S., Barrozo, M.A.S.: Optimization of hydrocyclone performance using multi-objective firefly colony algorithm. Separ. Sci. Technol. 48(12), 1891–1899 (2013)
26. Tilahun, S.L., Ong, H.C.: Modified firefly algorithm. Journal of Applied Mathematics 2012, Article ID 467631, 12 pages (2012)
27. Yang, X.-S.: Firefly algorithms for multimodal optimization. In Stochastic Algorithms: Foundations and Applications (SAGA 2009) Watanabe, O., Zeugmann, T. (eds.) Lect. Notes Comput. Sc. vol. 5792, 169–178 (2009)
28. Yang X.-S.: Firefly algorithm. In Nature-Inspired Metaheuristic Algorithms, 2nd edition, 81–96, Luniver Press, University of Cambridge, UK (2010)
29. Yang X.-S.: Multiobjective firefly algorithm for continuous optimization. Eng. Comput. 29(2), 175–184 (2013)
30. Yang X.-S., He, X.: Firefly algorithm: recent advances and applications. Int. J. Swarm Intelligence 1(1), 36–50 (2013)
31. Younes, M., Khodja, F., Kherfane, R.L.: Multi-objective economic emission dispatch solution using hybrid FFA (firefly algorithm) and considering wind power penetration. Energy 67, 595–606 (2014)
32. Yu, S., Yang, S., Su, S.: Self-adaptive step firefly algorithm. Journal of Applied Mathematics 2013, Article ID 832718, 8 pages (2013)
33. Zhou, A., Qu, B.-Y., Li, H., Zhao, S.-Z., Suganthan, P.N., Zhangd, Q.: Multiobjective evolutionary algorithms: A survey of the state of the art. Swarm and Evolutionary Computation 1(1), 32–49 (2011)
34. Zitzler, E., Deb, K., Thiele, L.: Comparison of multiobjective evolutionary algorithms: empirical results. Evolut. Comput. 8(2), 173–195 (2000)
35. Zitzler, E., Laumanns, M., Bleuler, S.: A tutorial on evolutionary multiobjective optimization. In Metaheuristics for Multiobjective Optimisation, Gandibleux, X., Sevaux, M., Sörensen, K., T'kindt, V. (eds.), Lect. Notes Econ. Math. vol. 535, 3–37 (2004)
36. Zitzler, E., Laumanns, M., Thiele, L.: SPEA2: Improving the strength Pareto evolutionary algorithm for multiobjective optimization. In Evolutionary Methods for Design, Optimisation and Control with Applications to Industrial problems, IC-NME, 95–100 (2002)