Engineering Applications of Artificial Intelligence 63 (2017) 20–32

Contents lists available at ScienceDirect

Engineering Applications of Artificial Intelligence

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

Non-dominated sorting moth flame optimization (NS-MFO) for multi-objective problems

Vimal Savsani [a,b], Mohamed A. Tawhid [b,c,*]

[a] Pandit Deendayal Petroleum University, Gandinagar, Gujarat, India
[b] Department of Mathematics and Statistics, Faculty of Science, Thompson Rivers University, Kamloops, BC, Canada V2C 0C8
[c] Department of Mathematics and Computer Science, Faculty of Science, Alexandria University, Moharam Bey 21511, Alexandria, Egypt

ARTICLE INFO

Article history:
Received 19 September 2016
Received in revised form
7 March 2017
Accepted 25 April 2017
Available online 8 May 2017

Keywords:
Multi-objective optimization
Moth-flame optimization algorithm
Multi-objective engineering design problems

ABSTRACT

This paper proposes an effective non-dominated moth-flame optimization algorithm (NS-MFO) method for solving multi-objective problems. Most of the multi-objective optimization algorithms use different search techniques inspired by different optimization techniques such as genetic algorithms, differential evolutions, particle swarm optimization, cuckoo search etc., but search techniques of recently developed metaheuristics have hardly been investigated. Non-dominated moth-flame optimization algorithm (NS-MFO) is based on the search technique of moth-flame optimization algorithm (MFO) algorithm and utilizes the elitist non-dominated sorting and crowding distance approach for obtaining different non domination levels and to preserve the diversity among the optimal set of solutions respectively. The effectiveness of the method is measured by implementing it on multi-objective benchmark problems and multi-objective engineering design problems with distinctive features. It is shown in this paper that this method effectively generates the Pareto front and also, this method is easy to implement and algorithmically simple.

1. Introduction

The solution of problems with only one objective function is a straightforward task. The output of such problems is the optimal solution and the algorithms are judged based on the quality of the optimal solutions. In multi-objective problems there are more than one objective functions, which are conflicting with each other, and there is no single optimal solution that simultaneously optimizes all the objective functions. In these cases, the decision makers are looking for the favored solution, rather than the optimal solution. These types of solutions are referred to as the Pareto solutions. So, in multi-objective optimization problems the insight of optimality is replaced with that of Pareto optimality (Mavrotas, 2009). The Pareto optimal (also referred as efficient, non-dominated, non-inferior) solutions are the solutions that cannot be improved in one objective function without weakening their performance in at least one of the rest. The set of the Pareto optimal solutions is known as the Pareto set. The general multi-objective problems are formulated as (Bérubé et al., 2009, Laumanns et al., 2006):

$$\min f(\vec{x}) = \left[f_1(\vec{x}),\quad f_2(\vec{x}),\ldots f_n(\vec{x})\right],\quad \vec{x} = \{x_1, x_2,\ldots,x_k\}\in S$$

$$\textit{Subjected to}: g_i(\vec{x})\le 0,\quad i = 1, 2, \ldots, I$$

$$h_j(\vec{x}) = 0,\quad j = 1, 2, \ldots J$$

$$(\vec{x})^l \le \vec{x} \le (\vec{x})^u$$

Where, S is the set of solutions referred to as solution space, $f_1(\vec{x}), f_2(\vec{x}),\ldots f_n(\vec{x})$ are individual objective functions, $g_i(\vec{x})$ are inequality constraints and $h_j(\vec{x})$ are equality constraints, $(\vec{x})^l$ *and* $(\vec{x})^u$ are bound constraints indicating lower and upper limits for the design variables. No single solution is possible that optimizes all the objectives simultaneously, especially when objective functions are conflicting with each other. This leads to Pareto-optimal solutions. The set of all Pareto-optimal design variables is called Pareto set. It can be noted that the *PS* is defined on the solution space whereas *PF* is defined on the objective space.

In recent time, the multi objective evolutionary algorithms (MOEAs) have gained enormous attention in solving the

* Corresponding author at: Department of Mathematics and Statistics, Faculty of Science, Thompson Rivers University, Kamloops, BC, Canada V2C 0C8.
E-mail addresses: vsavsani@tru.ca, Vimal.savsani@gmail.com (V. Savsani), mtawhid@tru.ca (M.A. Tawhid).

http://dx.doi.org/10.1016/j.engappai.2017.04.018
0952-1976/© 2017 Elsevier Ltd. All rights reserved.

engineering optimization problems with more than one objective. The multi/many objective optimization problems (MOOPs) differ from their single objective counter parts both in terms of their problem definitions/statements and methods to solve such problems. Weighted sum method, ε constraint method and such others for solving MOOPs are limited to the problems with simple objective functions and constraints. This is due to their nature to get stuck to a sub-optimal solution, and their unsuitability in solving a large variety of optimization problems (Deb, 2001). Alternatively, evolutionary algorithms (EAs), nature inspired algorithms and swarm intelligent algorithms have had a remarkable success in finding the global/near global optimum of complex problems nearly in all the disciplines of the knowledge. This success of such algorithms and their nature of using a population of solutions had led the researchers in employing it to optimize MOOPs. Such algorithms to solve MOOPs are referred as multi objective evolutionary algorithms (MOEAs) when they use EAs as their basic search techniques and are referred as simply multi objective optimization algorithms (MOOAs) when they use any metaheuristics in general. Primarily, all the methods to solve MOOPs have two goals to attain. The first is to find the Pareto front solutions as close as possible to the optimal Pareto front and the second is to maintain diversity among the optimal set of solutions.

One of the oldest attempt to have employed the EA to form a population based multi objective evolutionary algorithm was introduced in (Schaffer, 1985) and the method is known as vector evaluated genetic algorithms (VEGA). VEGA does the selection for each objective separately (Zitzler and Thiele, 1999), but its incapability in finding all the Pareto front solutions had limited its applications to very few real world optimization problems. These demerits of VEGA were overcome by algorithms which were inspired by the work of Goldberg (Goldberg and Holland, 1988), such as, multi objective genetic algorithm (MOGA) (Knowles and Corne, 1999), non-dominated sorting genetic algorithm (NSGA) (Srinivas and Deb, 1994), and niched Pareto genetic algorithm (NPGA) (Corne et al., 2001). These new algorithms introduced the concept of Pareto optimality into the selection process which awarded them a great success in application to various disciplines of engineering as thoroughly described in (Li and Zhang, 2009). Consequently the researchers had started experimenting various ways in assigning fitness values to the populations and in maintaining the diversity of optimal points which led to the development of new state of the art MOEAs for example strength Pareto evolutionary algorithm (SPEA2 and SPEA) (Zhou et al., 2011) and by (Zitzler and Thiele, 1999), Pareto archived evolution strategy (PAES) (Knowles and Corne, 1999), Pareto envelope-based selection algorithm (PESA and PESA-II) (Corne et al., 2000) and (Corne et al., 2001), an elitist based non-dominated sorting genetic algorithm – II (NSGA-II) (Deb et al., 2002), a multi objective evolutionary algorithm based on decomposition (MOEA/D) (Zhang and Li, 2007) and its improved versions described in detail in (Zhou et al., 2011). In addition, swarm intelligence based search algorithms for multi objective optimization (Coello and Lechuga, 2002; Mostaghim and Teich, 2004; Agrawal et al., 2008) have also been developed and applied to a variety of MOOPs.

Nature based meta-heuristics have caught the attention of different researchers due of its capability to solve problems which are multi-dimensional, multi-modal, combinatorial or large search space problems. The concept of nature based meta-heuristic was laid down by Holland in 1975 (Holland, 1975) with the introduction of genetic algorithm (GA), after that many population based effective algorithms were developed in next three decade like ant colony optimization (ACO) (Dorigo, 1992), artificial immune algorithm (AIA) (Farmer et al., 1986), differential evolution (DE) (Storn and Price, 1997), bacteria foraging algorithm (BFA) (Passino, 2002), shuffled frog leap (SFL) (Eusuff and Lansey, 2003) and particle swarm optimization (PSO) (Kennedy and Eberhart, 1995). These algorithms were applied to various engineering, scientific and management real life problems. The effectiveness of such algorithms has motivated researchers further to develop new algorithms based on the principle of nature. The development of nature based algorithms was remarkable since 2005 and many new algorithms were developed after that period. It is difficult to mention all, but few such algorithm includes, The invasive weed optimization (IWO) (Mehrabian and Lucas, 2006), artificial bee colony (ABC) (Karaboga and Basturk, 2007), biogeography-based optimization (BBO) (Simon, 2008), gravitational search algorithm (GSA) (Rashedi et al., 2009), grey wolf optimization (GWO) (Mirjalili et al., 2014), firefly algorithm (FA) (Yang, 2009), cuckoo search (CS) algorithm (Yang and Deb, 2010), bat algorithm (BA) (Yang, 2010), grenade explosion method (GEM) (Ahrari and Atai, 2010), charged system search (CSS) (Kaveh, 2014), teaching-learning based optimization (TLBO) (Rao et al., 2011, 2012), The animal migration optimization (AMO) (Li et al., 2014), water cycle algorithm (WCA) (Eskandar et al., 2012), runner root algorithm (RRA) (Merrikh-Bayat, 2015), mine blast algorithm (MBA) (Sadollah et al., 2013), Heat transfer search (HTS) (Patel and Savsani, 2015), Lightning search algorithm (LSA) (Shareef et al., 2015), lion optimization algorithm (LOA) (Yazdani and Jolai, 2015), Monarch butterfly optimization (MBO) (Wang et al., 2015a, 2015b), Earthworm optimization algorithm (EOA) (Wang et al., 2015a, 2015b), Passing Vehicle Search (PVS)(Savsani and Savsani, 2016), Elephant herding optimization (EHO) (Wang et al., 2016), Moth search algorithm (MSA)(Wang, 2016) and many more. Many of these algorithms have proven their ability for the applications on real life applications.

Many of these algorithms have their effective multi-objective versions like GA (Deb, 2001; Coello et al., 2002), PSO (Reyes and Coello, 2006; Moslehi and Mahnam, 2011; Wang and Yang, 2009), ABC (Omkar et al., 2011; Akbari et al., 2012a; Zhang et al., 2012), ACO (Angus and Woodward, 2009; Yagmahan and Yenisey, 2008); AIA (Aydin et al., 2011; Gong et al., 2008), GSA (Mondal et al., 2013), BBO (Jamuna and Swarup, 2012; Roy et al., 2010), IWA (Nikoofard et al., 2012), FFA (Yang, 2013), CSA (Yang and Deb, 2013), BA (Yang, 2011) and TLBO (Krishnanand et al., 2011; Patel and Savsani, 2014a, 2014b).

Moth-flame optimization algorithm (MFO) is an effective algorithm for the single objective optimization (Mirjalili, 2015) which works on the spiral movement of the moths around light source/flames. This method has proved its effectiveness and robustness in terms of accuracy, computational efforts and convergence for various applications. MFO is applied to solve challenging engineering problems (Jangir et al., 2016) whereas, Tamany et al. (2015) has demonstrated MFO for the training of multilayer ANN. MFO is also successfully applied to multi constrained Economic Order Quantity (EOQ) model by Khalilpourazari and Pasandideh (2017). MFO is also applied for parameter extractions of a multi-crystalline solar cell model (Allam et al., 2016) and optimal power flow was demonstrated using MFO by Bentauati et al. (2016).

This paper is mainly focused on to develop a technique by using multi-objective optimization method along with a recently developed meta-heuristics called Moth-flame optimization algorithm (MFO). This proposed method will be denoted as NS-MFO method. Later, in this paper it is shown that the NS-MFO is capable of generating non-dominated solutions and true/approximate Pareto front (PF). The first contribution of this paper is to introduce a new concept of NS-MFO algorithm and second contribution is to show the correctness of NS-MFO method for benchmark multi-objective problems and engineering design problems. The proposed approach is compared with the other well-known multi-objective optimization algorithms.

The paper is organized as follows: Section 2 describes the Moth-flame optimization algorithm (MFO) algorithm in details. Section 3 is focused to cover the basic structure of NS-MFO algorithm. Sections 4 and 5 investigates the performance of NS-MFO on different benchmark problems followed by Section 6 which investigates the performance of NS-MFO on different engineering design problems. Section 7 concludes the major findings of this work.

2. Moth-flame optimization algorithm(MFO)

Moth-flame is a nature inspired optimization method which works on the interesting spiral movement of moth in the night around a light source. This movement is also called transverse movement in which it maintains a fixed angle with respect to the light source. This movement of the moth is modeled mathematically and used for the optimization of a function by using moth-flame optimization (MFO) algorithm (Mirjalili, 2015). MFO is basically a population based method which starts the search with the initialization of random set of solutions in the search space, generally referred as population of moths. MFO updates the position of moths with respect to flames. Flames in MFO also refer to a set of solutions, but differ with respect to moth in a way that it only considers the best position of the moth achieved so far. Also, the size of the flames decreases as the algorithm progresses. The size of the flame at the start of the algorithm is same as the population size, but as the number of generation increases it decreases by following expression:

$$FS = round\left(F_{max} - G_c(F_{\max} - 1)/\left(G_{\max}\right)\right.$$

Where FS indicates the flame size, $F_{\max}$ is the maximum number of flames, $G_c and G_{\max}$ represent current and maximum number of generations.

The position of the moth is updated by using the following expression:

$$M_i = \left|F_j - M_i\right|e^{bt}\cos\left(2\pi t\right) + F_j$$

Where, M_i is the ith moth and F_j is the jth flame, b is the constant to define the shape of the spiral and t is the random number between -1 and 1. The MFO can be easily understood from Algorithm-1.

Algorithm 1. Moth-flame optimization algorithm

```
Create random population of moth and calculate the objective
  function
Create a set of flames (initially same as moth solution)
Update the position of the moths
Change the flame size
Return best solution
```

3. Non-dominated moth-flame optimization algorithm (NS-MFO)

Elitist non-dominated sorting method and diversity preserving crowding distance approach of NSGA-II are introduced in the proposed NS-MFO algorithm for sorting of the population in different non-domination levels with computed crowded distance. An elitist non-dominated sorting for obtaining different non domination levels is described first and then crowding distance approach to preserve the diversity among the optimal set of solutions has been explained.

Firstly, for each solution obtained from the basic search method (i.e. MFO) or from initially generated random population P_o, all the objectives from the objective vector F are evaluated. In addition, a domination count n_p defined as number of solutions dominating the solution p and S_p which is a set of solutions dominated by solution p are calculated. Secondly, all the solutions p are assigned a domination count zero and are put in first non-dominated level also known as Pareto Front (PF) and their non-domination rank $\left(NDR_p\right)$ is set to 1. Thirdly, for each solution p with n_p=0, each member q of the set S_p is visited and its domination count n_q is reduced by one. While reducing n_q count if it falls to zero the corresponding solution q is put in second non-domination level and NDR_q is set to 2. The procedure is repeated for each member of second non-domination level to obtain the third non-domination level, and subsequently the procedure should be repeated until the whole population is sorted into different non-domination levels.

In crowding distance approach for maintaining diversity among the obtained solutions firstly the population is sorted according to value of each objective function in ascending order. An infinte crowding distance is then assigned to the boundary solutions, $i=1$ and $i= l$, of each objective. Here l is the total number of solutions in a particular non-dominated set. The boundary solutions are the minimum ($i=1$) and maximum ($i=l$) function values. Except the boundary solutions all the other solutions of the sorted population ($i=2$ *to* l-1) for each objective j ($j=1,2,\ldots,m$) are assigned the crowding distance (d_m^i) as:

$$d_j^i = \frac{f_j^{i+1} - f_j^{i-1}}{f_j^{\max} - f_j^{\min}}$$

In the above equation the righ hand side term is the difference in values of objective function j for two neighbouring solutions ($i+1$ and $i-1$) of solution i. Once all the solutions in sorted population of a particular non-dominated set are assigned the crowding distance then each solution i is assigned two entities, non-domination rank NDR_i and crowding distance CD_i. A crowded comparison operator ($\prec_n$) is used as follows to compare two solutions (i and j) as follows $i\prec_n j$, if ($NDR_i < NDR_j$) or (($NDR_i = NDR_j$) and ($CD_i > CD_j$)). This is, between two solutions the one with the lower non-domination rank is prefered and if both the solutions have same non-domination rank then one with the higher crowding distance is prefered.

Algorithm-2. NS-MFO

```
Generate $P_o$ randomly in S and evaluate F for the generated $P_o$
Sort the $P_o$ based on the elitist non dominated sort method and
  find the NDR and fronts
Compute CD for each front
Update solutions ($P_j$) using MFO algorithm
Merge $P_o$ and $P_j$ to create $P_i = P_o \cup P_j$
For $P_i$ perform step 2
  Based on NDR and CD sort$P_i$
Replace $P_o$ with $P_i$ for first $N_{pop}$ members of $P_i$
```

The procedure of the proposed NS-MFO algorithm has been shown in Algorithm-2. Firstly, initialize parameters such as population size $\left(N_{pop}\right)$, termination criteria, here it is maximum number of generation $\left(N_{gen}\right)$ to run the algorithm. Secondly, a random parent population P_o in feasible region S is generated and each objective function of the objective vector F for P_o is evaluated. Next, elitist based non-dominated sorting and crowding distance computation as explained in earlier section is applied on P_o. Thirdly, MFO algorithm is employed to create the new population P_j, which is then merged with P_o to form the merged population P_i. This P_i is sorted based on elitism non-domination, and based on the computed values of *NDR* and *CD* the best N_{pop} solutions are updated to form a new parent population. This process is repeated till the maximum umber of generations (iterations) are reached. It should be noted that the same algorithm can also be used with the termination criteria set on the basis of number of function evaluations. Since non-dominated sorting and crowding distance assignment of NS-MFO are adopted from NSGA-II the computational complexity of NS-MFO is also same as NSGA-II which is $O\left(m\left(N_{pop}\right)^2\right)$, where m is the total number of objective functions and N_{pop} is the population size.

4. Numerical examples

Numerical experiments are carried out for different multi-objective benchmark problems. The multi-objective benchmark functions considered in this section are SCH, ZDT1, ZDT2, ZDT3, and LZ (Schaffer, 1985; Zitzler et al., 2000; Li and Zhang, 2009) which possesses different characteristics. SCH is a one dimensional problem with convex Pareto front. ZDT1 have convex Pareto front, whereas ZDT2 have non-convex Pareto front. The Pareto front of ZDT3 have discontinuous Pareto front. The objective functions in LZ function are multi-modal and it adds additional challenge for the algorithm to find the true Pareto front. The performance of the algorithm is measured by following criteria in this section (Deb, 2001).

4.1. Generational distance (GD)

GD is calculated between the true Pareto front (PF^t) and the obtained Pareto front (PF^O). GD indicates the average distance between the obtained Pareto front and the true Pareto front. GD is defined as:

$$GD = \frac{\left(\sum_{p=1}^{k}(d_i)\right)}{k}$$

$$\text{Where,}\quad d_i = \left(\sum_{i=1}^{n}\left(PF_{i,p}^{O} - PF_{i,p}^{t}\right)^2\right)^{1/2}$$

Where, k is the number of Pareto Solutions, n is the number of objective functions, $PF_{i,p}^{o}$ indicate the p^{th} obtained Pareto solution for the i^{th} objective and $PF_{i,p}^{t}$ indicate the nearest point on true Pareto front from $PF_{i,p}^{o}$

4.2. Spacing (S)

Spacing is defined as:

$$S = \frac{1}{k-1}\sum_{p=2}^{k}\left(D_p - \bar{D}\right)^2$$

Where, D_i is the absolute difference between the two consecutive solutions in the obtained Pareto front (PF^o). It is defined as:

$$D_p = \sum_{i}^{n}\left|PF_{i,(p-1)}^{o} - PF_{i,p}^{o}\right|$$

$\bar{D}$ is the average of all D_p.

Spacing(S) specify the spread of the obtained Pareto front. It gives the standard deviation of D_g. The small value of S indicates the uniform spacing of the obtained Pareto Solutions.

4.3. Spread (Δ)

Spread is defined as:

$$\Delta = \frac{\sum_{j=1}^{n} d_j^{ex} + \sum_{p=2}^{k}\left|d_p - \bar{d}\right|}{\sum_{j=1}^{n} d_j^{ex} + (k-1)\bar{d}}$$

Where, d_p is the Euclidian distance between two consecutive points of the obtained Pareto front and d_j^{ex} is the Euclidian distance between the obtained Pareto front and the True Pareto front. $\bar{d}$ is the average of d_p. Spread checks the condition for the obtained Pareto front to cover the True Pareto front. Smaller value of Δ indicate better spread for the obtained Pareto front with uniform distribution.

For the experimentation, the population size is set equal to the Pareto solutions required in PF and maximum numbers of generations are set as 500 for ZDT1, ZDT2, ZDT3 and SCH functions. For LZ function, number of generations is set to 1000, to maintain the uniformity of experimental setup. For all the problems number of Pareto Solutions (k) are considered as 200. The Pareto front for SCH, ZDT1, ZDT2, ZDT3 and LZ are show in Figs. 1–5 respectively. All the figures show the True Pareto front and the obtained Pareto front. It can be noted from the results that for all the functions NS-MFO is capable to approximate the True Pareto front. The effectiveness of the results is checked by the above mentioned performances measures. The results for the GD for all the functions are summarized in Table 1, where the results are compared with the state-of-the art multi-objective algorithms such as NSGAII, SPEA, VEGA, MODE, DEMO and MO-Bees. The results except NS-MFO are reproduced from Tripathi et al. (2007), Yang (2013), and Yang and Deb (2013). The results summarized in Table 1 are the average result obtained in 25 independent runs.

It is observed from the results that for SCH function NS-MFO has produced better GD then all the rest of the algorithms. For ZDT1, ZDT2, NS-MFO is better compared to all the algorithms. For ZDT3, it is only inferior to DEMO. For LZ, NS-MFO is inferior to VEGA, SPEA and DEMO. It can be summarized that NS-MFO is capable of finding true Pareto solutions for the problems with convex, non-convex and discrete Pareto front. The results are also

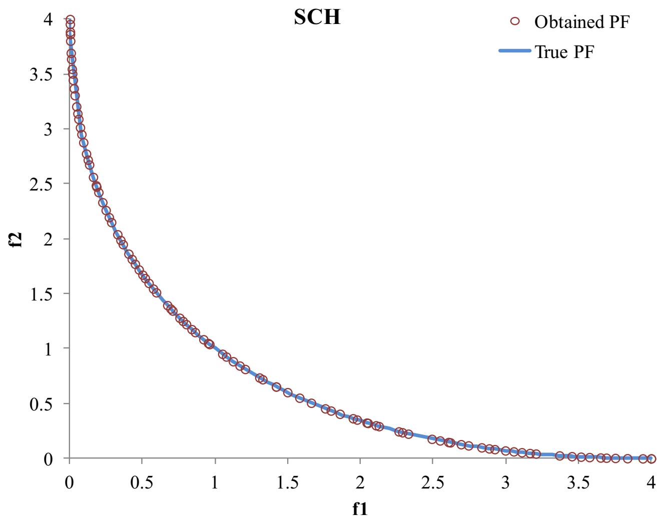

Fig. 1. Pareto front for SCH function.

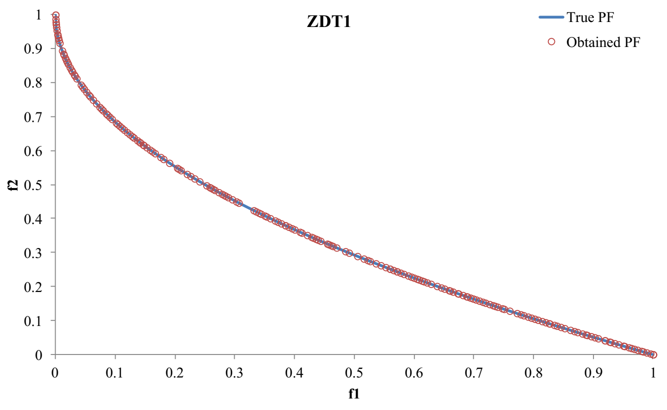

Fig. 2. Pareto front for ZDT1 function.

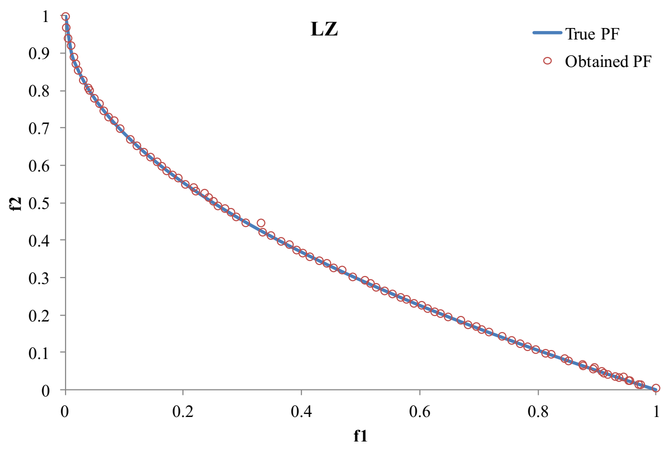

Fig. 5. Pareto front for LZ function.

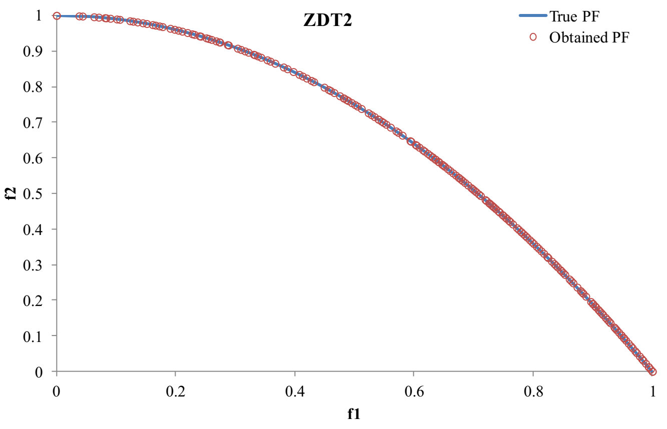

Fig. 3. Pareto front for ZDT2 function.

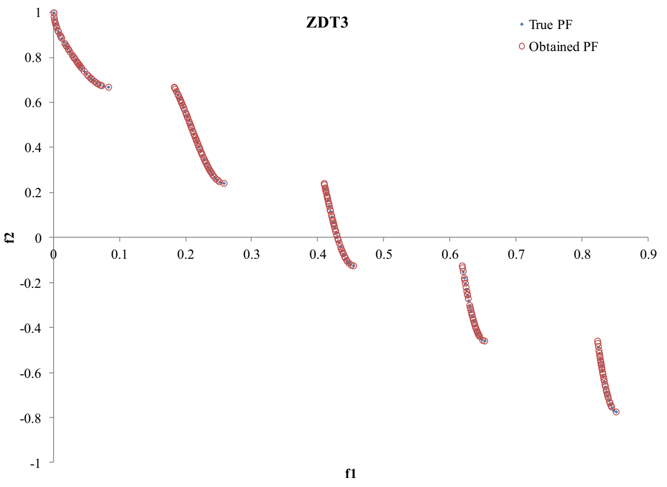

Fig. 4. Pareto front for ZDT3 function.

Table 1
Results(GD) for benchmark multi-objective problems.

	SCH	ZDT1	ZDT2	ZDT3	LZ
VEGA	6.98E−02	3.79E−02	2.37E−03	3.29E−01	1.47E−03
NSGA-II	5.73E−03	3.33E−02	7.24E−02	1.14E−01	2.77E−02
MODE	9.32E−04	5.80E−03	5.50E−03	2.15E−02	3.19E−03
DEMO	1.79E−04	1.08E−03	7.55E−04	1.18E−03	1.40E−03
Bees	1.25E−02	2.40E−02	1.69E−02	1.91E−01	1.88E−02
SPEA	5.17E−03	1.78E−03	1.34E−03	4.75E−02	1.92E−03
NS-MFO	6.86E−05	1.45E−05	6.29E−06	2.82E−03	2.22E−03

Table 2
Results (Δ) for the multi-objective benchmark problems.

	ZDT1	ZDT2	ZDT3
NSGA-II(RC)	0.3903	0.4307	0.7385
NSGA-II(BC)	0.4632	0.4351	0.5756
SPEA	0.7301	0.6781	0.6657
PAES	1.2297	1.1659	0.78992
PESAII	0.84816	0.89292	1.22731
σ-MOPSO	0.39856	0.38927	0.76016
NSPSO	0.90695	0.92156	0.62072
MOPSO	0.68132	0.63922	0.83195
NS-MFO	0.2431	0.2343	0.5945

compared based on the spread (Δ) with the other multi-objective optimization algorithms such as NSGA-II (real coded), NSGA-II (binary coded), SPEA, PAES, PESAII, σ-MOPSO, NSPSO, and MOPSO. The result for the Δ are shown in Table 2. It can be noted from the results that NS-MFO has shown better Δ for ZDT1 and ZDT2, while for ZDT3 it has produced nearly same to the NSGA-II(BC) and better than other algorithms.

5. Benchmark examples

Several benchmark functions are provided by Congress on evolutionary computation (CEC) for testing performance parameters of optimization algorithms. Benchmark functions for multi-objective optimization are provided in CEC 2009 (Zhang et al., 2008). The effectiveness of NS-MFO algorithm is evaluated by testing it on 20 well defined benchmark functions of CEC 2009 from which 10 functions are for multi-objective unconstrained problems (UF1-UF10) and the remaining 10 are for multi-objective constrained functions (CF1-CF10). The detailed mathematical formulations of the considered test functions are given in Zhang et al. (2008). The performance for the algorithms is evaluated by inverted generational distance (IGD) measure. The value of IGD indicates the distance of the approximate Pareto front from the true Pareto front. A low value of IGD indicates that the obtained Pareto front is closely approached to the true Pareto front. The IGD measure is defined as: Let P^* be a set of uniformly distributed points along the true Pareto front in the objective space. Let A be a set of points representing obtained Pareto front. Then the average

Table 3
Results(mean IGD) for unconstrained multi-objective benchmark problems of CEC 2009.

A[a]	UF1	UF2	UF3	UF4	UF5	UF6	UF7	UF8	UF9	UF10
1	4.21E−03	7.62E−03	6.72E−02	3.29E−02	6.29E−02	4.54E−02	2.02E−02	6.29E−02	2.00E−01	4.33E−01
2	8.79E−03	7.84E−03	6.93E−02	7.01E−02	9.79E−02	5.53E−02	6.80E−02	9.93E−02	1.64E−01	1.79E−01
3	4.35E−03	6.79E−03	7.42E−03	6.39E−02	1.81E−01	5.87E−03	4.44E−03	5.84E−02	7.90E−02	4.74E−01
4	6.46E−03	6.15E−03	5.31E−02	2.36E−02	1.49E−02	5.92E−02	4.08E−02	1.13E−01	1.14E−01	1.53E−01
5	1.04E−02	6.79E−03	3.34E−02	4.27E−02	3.15E−01	6.67E−02	1.03E−02	6.84E−02	4.90E−02	3.22E−01
6	6.18E−03	4.84E−03	5.12E−02	5.80E−02	7.78E−02	6.54E−02	5.57E−02	6.73E−02	6.15E−02	1.95E−01
7	7.85E−03	1.23E−02	1.50E−02	4.35E−02	1.62E−01	1.76E−01	7.30E−03	8.24E−02	9.39E−02	4.47E−01
8	6.20E−03	6.40E−03	4.29E−02	4.76E−02	1.79E+00	5.56E−01	7.60E−03	2.45E−01	1.88E−01	5.65E−01
9	5.34E−03	1.20E−02	1.06E−01	2.65E−02	3.93E−02	2.51E−01	2.52E−02	2.49E−01	8.25E−02	4.33E−01
10	7.70E−02	2.83E−02	9.35E−02	3.39E−02	1.67E−01	1.26E−01	2.42E−02	2.16E−01	1.41E−01	3.70E−01
11	2.99E−02	2.28E−02	5.49E−02	5.85E−02	2.47E−01	8.71E−02	2.23E−02	2.38E−01	2.93E−01	4.11E−01
12	3.59E−02	1.62E−02	7.00E−02	4.06E−02	9.41E−02	1.29E−01	5.71E−02	1.71E−01	1.89E−01	3.24E−01
13	8.56E−02	3.06E−02	2.71E−01	4.62E−02	1.69E−01	7.34E−02	3.35E−02	1.92E−01	2.32E−01	6.28E−01
14	1.22E−02	8.10E−03	1.03E−01	5.13E−02	4.30E−01	1.92E−01	5.85E−02	9.45E−02	9.83E−02	7.43E−01
15	5.96E−02	1.89E−02	9.90E−02	4.27E−02	2.25E−01	1.03E−01	1.97E−02	4.23E−01	3.42E−01	3.62E−01
16	1.15E−02	1.24E−02	1.06E−01	5.84E−02	5.66E−01	3.10E−01	2.13E−02	8.63E−02	7.19E−02	8.45E−01
17	5.79E−03	7.84E−03	6.93E−02	3.59E−02	3.79E−02	5.53E−02	2.14E−02	9.93E−02	1.17E−01	1.79E−01

[a] A-Algorithms: 1-Multi-objective moth flame optimization (NS-MFO), 2-Multi-objective teaching learning based optimization (MO-TLBO), 3-(MultiObjective Evolutionary Algorithm based on Decomposition (MOEA/D), 4- multiple trajectory search (MTS), 5- dynamical multiobjective evolutionary algorithm with decomposition technique (DMOEADD), 6- Multiobjective ABC (MOABC), 7- multiobjective evolutionary algorithm based on determined weight and sub-regional search (LiuLi Algorithm), 8- MOEA/D with Guided Mutation and Priority Update (MOEADGM), 9- Generalized Differential Evolution 3 (GDE3), 10- Differential Evolution with Self-adaptation and Local Search for Constrained Multiobjective Optimization algorithm (DECMOSA-SQP), 11- Clustering Multi-objective Evolutionary Algorithm (Clustering MOEA), 12- Archive-based Micro Genetic Algorithm (AMGA), 13- Orthogonal Multi-objective Evolutionary Algorithm (OMOEA-II), 14- Multiobjective Self-adaptive Differential Evolution algorithm with objective-wise learning strategies (OW-MOSaDE), 15- Multi-objective evolutionary programming (MOEP) using fuzzy rank-sum with diversified selection, 16- nondominated genetic algorithm with local search (NSGAIILS), 17-Multi-objective biogeography based optimization with ACO (MO-BBO_ACO).

distance from P^* to A is known as IGD and represent as follows.

$$IGD(A, P^*) = \frac{\sum_{\nu \in P^*} d(\nu, A)}{|P^*|}$$

Where, $d(\upsilon, A)$ is the minimum Euclidian distance between υ and the other points in A. If $|P^*|$ is large enough to represent the Pareto front very well, both diversity and convergence of the approximated set A could be measured using IGD (A, P^*). An optimization algorithm will try to minimize the value of IGD (A, P^*) measure. To obtain smaller values of this measure, the approximated set A must be very close to the true Pareto front and cannot miss any part of the whole Pareto front. In order to maintain the consistence in the comparison of the competitive algorithms, a common platform is provided by CEC 2009. The total number of function evaluations is set as 300,000 and population size is set as 50. The comparison is based on the average value of IGD obtained in 30 independent runs. The result of NS-MFO is compared with 16 other algorithms for unconstrained problems. These algorithms are derived from other optimization methods such as genetic algorithm, differential evolutions, artificial bee colony technique, biogeography based optimization or teaching learning based optimization techniques. The results of average IGD calculated from 30 runs are listed in Table 3 for 10 unconstrained multi-objective benchmark problems (UF1-UF10). The results of NS-MFO for constrained benchmark problems are compared with 8 other algorithms and the results of average IGD is listed in Table 5 All the results except NS-MFO are rewritten from Akbari et al. (2012b), Patel and Savsani (2016) and Savsani et al. (2014) for the comparison purpose. Friedman rank test is performed to rank the algorithms for all the unconstrained and constrained problems based on average IGD. Friedman rank value along with its normalized value and rank for unconstrained problems are given in Table 4 and for constrained problems are given in Table 6. It can be observed from the results that for unconstrained multi-objective problems NS-MFO has ranked 3rd among 17 algorithms and 2nd among 9 algorithms for constrained problems.

Table 4
Friedman rank test for unconstrained multi-objective benchmark problems of CEC 2009.

Algorithms	Friedman value	Normalized value	Rank
NS-MFO	56	0.96	3
MO-TLBO	92.5	0.58	8
MOEAD	54.5	0.99	2
MTS	54	1.00	1
DMOEADD	59.5	0.91	5
MOABC	58	0.93	4
LiuLi Algorithm	76	0.71	6
MOEADGM	102	0.53	9
GDE3	92	0.59	7
DECMOSA-SQP	109	0.50	10
Clustering MOEA	120	0.45	12
AMGA	107	0.50	10
OMOEAII	133	0.41	14
OW MOSaDE	122	0.44	13
MOEP	118	0.46	11
NSGAIILS	117	0.46	11
MO-BBO_ACO	59.5	0.91	5

6. Engineering design experiments

In this section NS-MFO is experimented on 6 different multi-objective engineering design problems which are widely used in the research to investigate the multi-objective optimization algorithms. All the considered problems possess different characteristics. The algorithms parameters for the NS-MFO are same as that for the benchmark problems considered in the previous section. All the engineering problems are experimented with a population size of 100 (solutions required in the Pareto front) and 500 iterations. For all the problems 100 Pareto solutions are generated and the results mentioned in this work are the average result obtained in the 30 runs to maintain the same experimental set up with (Sadollah et al., 2015).

6.1. Four bar truss problem

The aim of this problem is to minimize volume and displacement of joints simultaneously. Area of each link is considered as design

Table 5
Results(mean IGD) for constrained multi-objective benchmark problems of CEC 2009.

A[a]	CF1	CF2	CF3	CF4	CF5	CF6	CF7	CF8	CF9	CF10
1	6.34E−02	4.73E−03	4.55E−02	4.32E−03	5.23E−02	3.04E−02	2.62E−02	3.24E−02	7.23E−02	1.25E−01
2	2.49E−02	1.20E−02	9.80E−02	8.68E−03	7.64E−02	1.64E−02	3.33E−02	2.98E−01	8.10E−02	3.01E−01
3	1.13E−02	2.10E−03	5.63E−02	6.99E−03	1.58E−02	1.50E−02	1.91E−02	4.75E−02	1.43E−01	1.62E−01
4	8.50E−04	4.20E−03	1.83E−01	1.42E−02	1.10E−01	1.39E−02	1.04E−01	6.07E−02	5.05E−02	1.97E−01
5	1.92E−02	2.68E−02	1.04E−01	1.11E−02	2.08E−02	1.62E−02	2.47E−02	1.09E+00	8.51E−02	1.38E−01
6	1.08E−02	8.00E−03	5.13E−01	7.07E−02	5.45E−01	2.07E−01	5.36E−01	4.06E−01	1.52E−01	3.14E−01
7	1.08E−01	9.46E−02	1.00E+06	1.53E−01	4.13E−01	1.48E−01	2.60E−01	1.76E−01	1.27E−01	5.07E−01
8	2.94E−02	1.60E−02	1.28E−01	7.99E−03	6.80E−02	6.20E−02	4.17E−02	1.39E−01	1.15E−01	4.92E−01
9	6.92E−03	1.18E−02	2.40E−01	1.58E−02	1.84E−01	2.01E−02	2.33E−01	1.11E−01	1.06E−01	3.59E−01

[a] A-Algorithms: 1-Multi-objective moth flame optimization (NS-MFO), 2-Multi-objective teaching learning based optimization (MO-TLBO), 3- Dynamical multi-objective evolutionary algorithm with decomposition technique (DMOEADD), 4- Multi-objective evolutionary algorithm based on determined weight and sub-regional search (LiuLi Algorithm), 5- Multiple trajectory search (MTS), 6- MOEA/D with Guided Mutation and Priority Update (MOEADGM), 7- Differential Evolution with Self-adaptation and Local Search for Constrained Multi-objective Optimization algorithm (DECMOSA-SQP), 8- Generalized Differential Evolution 3 (GDE3), 9- Nondominated genetic algorithm with local search (NSGAIILS).

Table 6
Friedman rank test for constrained multi-objective benchmark problems of CEC 2009.

Algorithms	Friedman value	Normalized value	Rank
NS-MFO	29	0.90	2
MO-TLBO	47	0.55	5
DMOEADD	26	1.00	1
LiuLi Algorithm	36	0.72	3
MTS	44	0.59	4
MOEADGM	73	0.36	7
DECMOSA-SQP	82	0.32	8
GDE3	57	0.46	6
NSGAIILS	56	0.46	6

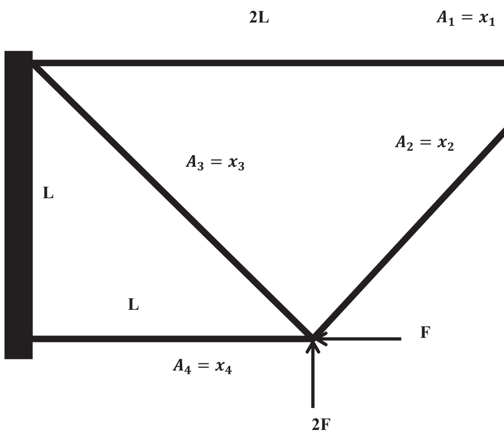

Fig. 6. Four bar truss problem.

variables. This problem is an unconstrained problem with all the continuous design variables. The system is shown in Fig. 6.

The problem can be stated as below

$$\text{Minimize}: f_1(x) = L\left(2x_1 + \sqrt{2x_2} + \sqrt{x_3} + x_4\right)$$

$$\text{Minimize}: f_2(x) = \left(F\frac{L}{E}\right)\left(\frac{2}{x_2} + 2\frac{\sqrt{2}}{x_2} - 2\frac{\sqrt{2}}{x_3} + \frac{2}{x_4}\right)$$

Where,

$$F = 10, \quad E = 2e5, \quad L = 200$$

Table 7
Result (S) for the four bar truss problem.

Algorithms	S	
	Mean	SD
NSGA-II	2.3635	0.2551
MOPSO	2.5303	0.2275
Micro-GA	8.2742	16.8311
PAES	3.2314	5.9555
MOWCA	2.5816	0.0298
NS-MFO	2.3844	0.2312

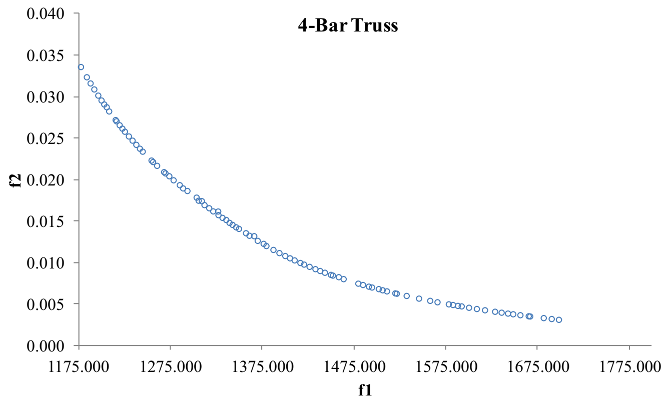

Fig. 7. Pareto front for the four bar truss problem.

$$1 \le x_1, \quad x_4 \le 3, \quad \sqrt{2} \le x_2, \quad x_3 \le 3$$

This problem has a convex Pareto front in the objective space. The extreme point found for this problem are (1174.200, 0.0341) and (1727.739, 0.00276). As mathematical expression for the exact Pareto front is not available for this problem in the literature the performance is checked based on the value of spacing (S) due to its availability in the literature for different algorithms. The value of S for NS-MFO along with different algorithms such as NSGA-II, MOPSO, Micro-GA, PAES and MOWCA (Sadollah et al., 2015) are given in Table 7. It can be observed from the results that NS-MFO has produced better or nearly same spacing compared to all the algorithms. The obtained Pareto front is presented in Fig. 7, which justifies the results presented in the Table 7. The obtained Pareto front matches well with the available Pareto front in the literature (Sadollah et al., 2015).

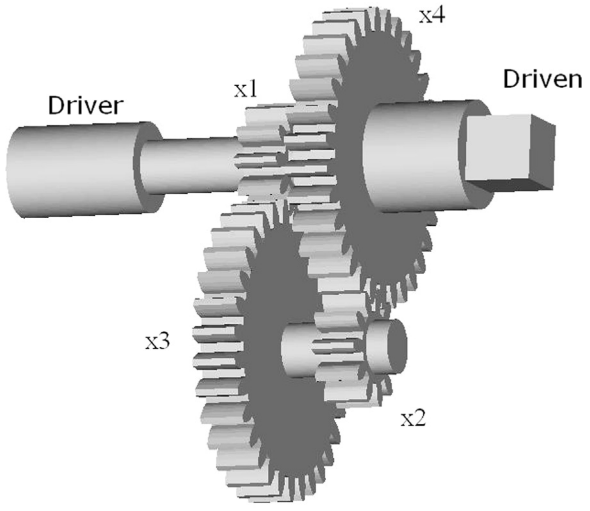

Fig. 8. Gear train.

6.2. Gear train problem

The purpose of this problem is to minimize the maximum size of any one of the gear simultaneously with the minimization of the gear ratio error with reference gear ratio of 1/6.931. All the design variables considered in this example can only take integer values as it represents the number of teeth on each gear. Hence, this problem is a min-max problem with discrete design variables, which offers additional challenges to an optimization algorithm. The system is shown in Fig. 8.

The problem can be stated as:

$$\text{Minimize } f_1(x) = \left(\left(\frac{1}{6.931}\right) - \left(\frac{x_1x_2}{x_3x_4}\right)\right)^2$$

$$\text{Minimize } f_2(x) = max([x_1x_2x_3x_4])$$

Where, $12 \leq x_1, x_2, x_3, x_4 \leq 60$

The performance measures such as GD, S and Δ are not available for the gear train problems. So, the results are compared based on the extreme Pareto solutions available in the literature. The results are compared with NSGA-II and MOWCA. The results are summarized in Table 8.

It can be observed from the results that the extreme Pareto solutions obtained by NS-MFO is (2.36e−9, 40) and (7.32e–1, 12). These extreme solutions are better than NSGA-II and upper extreme is slightly inferior to MOWCA. This problem has a discrete Pareto front and it is shown in Fig. 9, obtained by NS-MFO.

Table 8
Result (extreme pareto solutions) for the gear train problem.

Algorithms		Objective function	
		f_1	f_2
NSGA-II	$f_1 \rightarrow min$	1.83e−8	37
	$f_2 \rightarrow min$	5.01e−1	13
MOWCA	$f_1 \rightarrow min$	4.5e−9	43
	$f_2 \rightarrow min$	7.32e−1	12
NS-MFO	$f_1 \rightarrow min$	2.36e−9	40
	$f_2 \rightarrow min$	7.32e−1	12

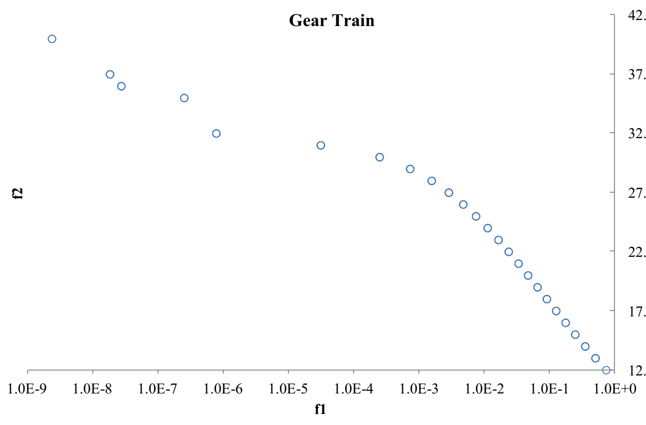

Fig. 9. Pareto front for gear train problem.

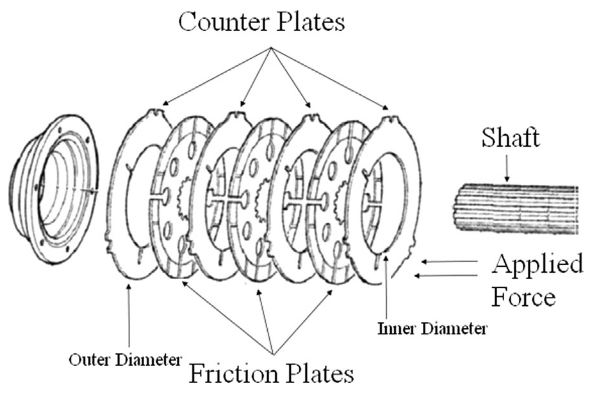

Fig. 10. Exploded view for the multi-plate disc brake.

6.3. Multi-plate disc brake design

Multi-plate disc brake finds its applications in airplanes, to apply effective braking while landing. The exploded view of the multi-pate disc brake is shown in Fig. 10. The purpose of the problem is to simultaneously minimize the mass of the brake and the stopping time. There are four design variables for the inner radius, outer radius, the engaging force (applied force) and the number of friction surfaces (number of friction plates). Out of these four design variables, number of friction surfaces can only take a discrete integer values which make it a mixed-integer problem. This is also a constrained problem for which five different restrictions are introduced for the distance between the radii of the friction plates, length of the brake, pressure sustained by the plates, maximum limitation for the temperature generated and the braking torque. The problem can be stated as follows:

$$\text{Minimize } f_1(x) = 4.9e - 5(x_2^2 - x_1^2)(x_4 - 1)$$

$$\text{Minimize } f_2(x) = (9.82e6)\frac{x_2^2 - x_1^2}{x_3x_4(x_2^3 - x_1^3)}$$

$$g_1 = 20 + x_1 - x_2$$

$$g_2 = 2.5(x_4 + 1) - 30$$

$$g_3 = \frac{x_3}{3.14(x_2^2 - x_1^2)^2} - 0.4$$

Table 9
Result (Δ) for the disc brake problem.

Algorithm	Δ	
	Mean	SD
NSGA-II	0.79717	0.06608
paε – ODEMO	0.8401	0.20085
MOWCA	0.46041	0.10961
NS-MFO	0.3845	0.2632

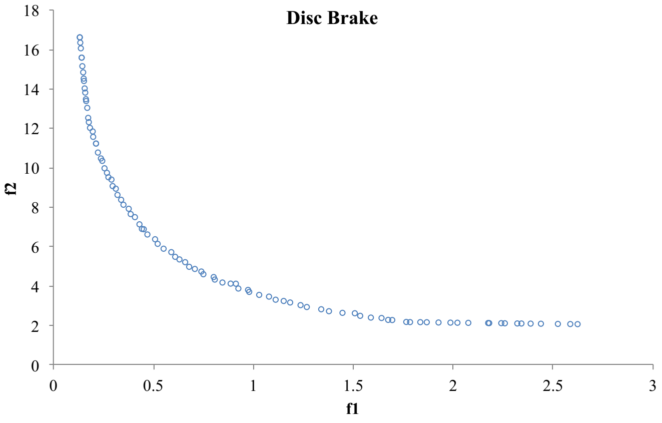

Fig. 11. Pareto front for disc brake problem.

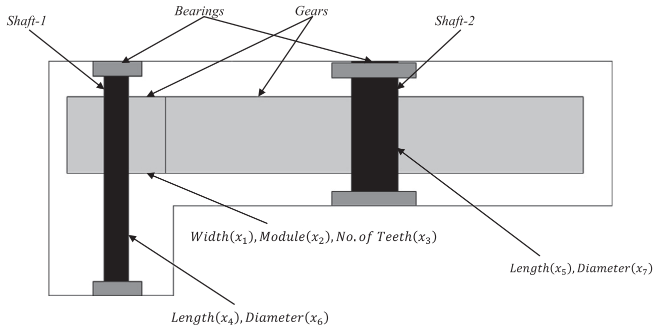
Shaft-1
Bearings
Gears
Shaft-2
$Width(x_1), Module(x_2), No. of Teeth(x_3)$
$Length(x_5), Diameter(x_7)$
$Length(x_4), Diameter(x_6)$

Fig. 12. Speed reducer.

$$g_4 = 2.\,22e{-}3x_3\frac{x_2^3 - x_1^3}{\left(x_2^2 - x_1^2\right)^2}{-}1$$

$$g_5 = 900{-}\left(\frac{2.\,66e{-}2x_3x_4\left(x_3^3 - x_1^3\right)}{x_2^2 - x_1^2}\right)$$

$\forall\ g\le 0$

$55 \le x_1 \le 80,\ 75 \le x_2 \le 110,\ 1000 \le x_3 \le 3000,\ 2 \le x_4 \le 20,\ x_4 \in I$

The extreme Pareto solutions obtained are (0.1274016.6549) and (2.7915, 2.07204). The results are compared for the value of spacing (Δ). The results are shown in Table 9, where the performance of NS-MFO is compared with NSGA-II, *paε – ODEMO* and MOWCA. It can be noted from the results that NS-MFO has produced better Δ compared to NSGA-II, *paε–ODEMO* and MOWCA. The Pareto front obtained by using NS-MFO is shown in Fig. 11, which is either same or better than the Pareto front given in

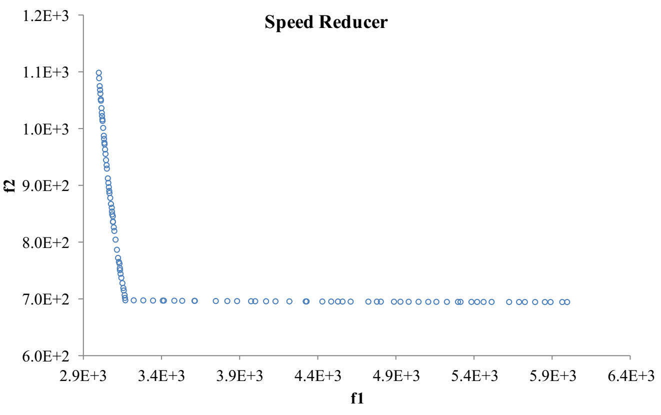

Fig. 13. Pareto front for speed reducer problem.

(Sadollah et al., 2015). It is useful for the designer to note that the Pareto solutions becomes nearly stagnant for f_1 at f_1 ≅0.2 and for f_2 at f_2≅1.75. (Fig. 12)

6.4. Speed reducer problem

The aim of this problem is to simultaneously optimize the weight of the gear assembly and the transverse deflection of the shaft. The assembly of the sped reducer is shown in Fig. 13. The problem is subjected to different constraints on bending stress of the gear teeth, surfaces stress, transverse deflections of the shafts and stresses in the shafts. The design variables are the face width, module of teeth, number of teeth in the pinion, length of the first shaft between bearings, length of the second shaft between bearings and the diameter of the first and second shafts respectively. All the variables are indicated in the Fig. 11. The third variable is integer, the rest of them are continuous. So this problem can be classified as a mixed-integer problem.

The problem can be stated as follows:

$$\begin{aligned} Minimize\ f_1 \;&=\; 0.7854\,x_1x_2^2\left(3.3333x_3^2{+}14.9334x_3{-}43.0934\right)\\ &{-}1.508x_1\left(x_6^2 + x_7^2\right) + 7.4777\left(x_6^3 + x_7^3\right)\\ &+ 0.7854\left(x_4x_6^2 + x_5x_7^2\right)\end{aligned}$$

$$Minimize\ f_2 = \frac{\sqrt{\left(745\frac{x_4}{x_2x_3}\right)^2{+}1.69e7}}{0.1{*}x_6^3}$$

$$g_1 = -\left(27/\left(x_1x_2^2x_3\right) - 1\right)$$

$$g_2 = -\left(397.5/\left(x_1x_2^2x_3^3\right) - 1\right)$$

$$g_3 = -\left(1.93\frac{x_4^3}{x_2x_3x_6^4}{-}1\right)$$

$$g_4 = -\left(1.93\frac{x_5^3}{x_2x_3x_7^4}{-}1\right)$$

$$g_5 = -\left(\frac{\sqrt{\left(745\frac{x_4}{x_2x_3}\right)^2{+}16.9e6}}{0.1x_6^3}{-}1\right)$$

Table 10
Result (S) for speed reducer problem.

Algorithms	S	
	Mean	SD
NSGA-II	2.765	3.534
Micro-GA	47.80	32.80
PAES	16.20	4.268
MOWCA	16.68	2.697
NS-MFO	19.3452	4.3321

$$g_6 = -\left(\frac{\sqrt{\left(745\frac{x_5}{x_2x_3}\right)^2+157.5e6}}{0.1x_7^3}-1\right)$$

$$g_7 = -\left(x_2*\frac{x_3}{40}-1\right)$$

$$g_8 = -\left(5*\frac{x_2}{x_1}-1\right)$$

$$g_9 = -\left(\frac{x_1}{12*x_2}-1\right)$$

$$g_{10} = -\left(\frac{1.5*x_6+1.9}{x_4}-1\right)$$

$$g_{11} = -\left(\frac{1.1*x_7+1.9}{x_5}-1\right)$$

$$\forall\, g1 \geq 0$$

$$2.6 \leq x_1 \leq 3.6,\ 0.7 \leq x_2 \leq 0.8,\ 17 \leq x_3 \leq 28,\ 7.3 \leq x_4, x_5 \leq 8.3,$$

$$2.9 \leq x_6 \leq 3.9,\ 5 \leq x_7 \leq 5.5$$

The extreme Pareto solutions obtained by NS-MFO are (2995, 1099.45) and (5884.31, 694.73). The results are compared based on the Spacing (S) with the other algorithms such as NSGA-II, Micro-GA, PAES and MOWCA (Sadollah et al., 2015). The results are presented in the Table 10.

It can be observed from the results that, the Spacing of NSGA-II better compared to all the algorithms , but the extreme points of the Pareto front obtained by NSGA-II is inferior compared to other algorithms (Sadollah et al., 2015). Also, value of S is inferior for NS-MFO to the other methods, but the obtained PF is quiet near to the actual PF. The Pareto front is given in Fig. 13. It is useful for the designer to note that the Pareto solutions becomes nearly stagnant for f_2 at $f_2 \cong 697$.

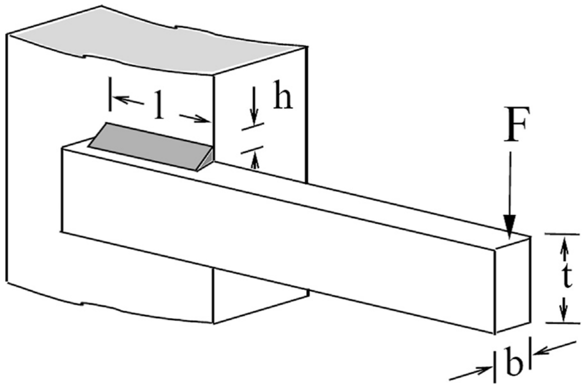

Fig. 14. Welded beam.

6.5. Welded beam design

The purpose of this problem is to minimize the cost simultaneously with the end deflection. The problem is subjected to the constraints on shear stress, bending stress, weld length and the buckling load. The four different design variables are the height and the length of the welded joint and thickness and the width of the beam. All the design variables are continuous. The schematic diagram is shown in Fig. 14, where all the variables are shown clearly. The value of objective functions differs much in its values and so such problems are difficult to solve with the weighted sum method. The problem can be stated as:

$$Minimize\ f_1 = 1.10471x_1^2x_2 + 0.04811x_3x_4(14+x_2)$$

$$Minimize\ f_2 = del$$

$$g_1 = -(tau - taumax)$$

$$g_2 = -(sig - sigmax)$$

$$g_3 = -(x_1 - x_4)$$

$$g_4 = -(P - pc)$$

Where,

$$pc = \left(\frac{4.013*E\left(\sqrt{\frac{x_3^2x_4^6}{36}}\right)}{L^2}\right)\left(1-\left(\left(\frac{x_3}{2L}\right)\sqrt{\frac{E}{4G}}\right)\right)$$

$$del = \frac{4PL^3}{Ex_3^3x_4}$$

$$sig = \frac{6PL}{x_4x_3^2}$$

$$J = 2*\left(\sqrt{2}x_1x_2\left(\left(\frac{x_2^2}{12}\right)+\left(\frac{x_1+x_3}{2}\right)^2\right)\right)$$

$$R = \sqrt{\left(\frac{x_2^2}{4}\right)+\left(\frac{x_1+x_3}{2}\right)^2}$$

$$M1 = P*\left(L+\frac{x_2}{2}\right)$$

Table 11
Results (Δ) for welded beam problem.

Algorithm	Δ	
	Mean	SD
NSGA-II	0.8898	0.1197
paε − ODEMO	0.5860	0.0436
MOWCA	0.2247	0.0928
NS-MFO	0.2337	0.0923

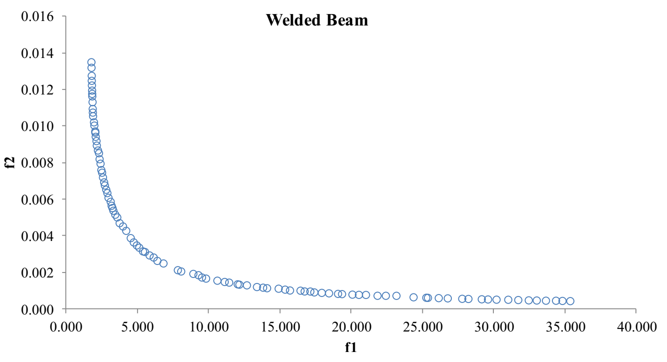

Fig. 15. Pareto front for welded beam problem.

$$tau2 = \frac{M1R}{J}$$

$$tau1 = \frac{P}{\sqrt{2}x_1x_2}$$

$$tau = \sqrt{tau1^2 + 2*tau1*tau2*\frac{x_2}{2*R} + tau2^2}$$

$$\forall\ g \geq 0$$

$P = 6000$, $L = 14$, $E = 30e6$, $G = 12e6$, $taumax = 13600$, $sigmax = 30000$,

$delmax = 0.25$

$0.125 \leq x_1, x_4 \leq 5, \quad 0.1 \leq x_2, x_3 \leq 10$

The extreme Pareto solutions obtained by NS-MFO are (1.74766,0.01351) and (35.33, 0.000439). This problem is compared with the other techniques such as NSGA-II, $pa\varepsilon - ODEMO$, and MOWCA (Sadollah et al., 2015). The results are compared based on Δ and the mean values along with the calculated standard deviation are summarized in Table 11. It can be observed from the results that NS-MFO possesses better spreading compared to the other algorithms. The Pareto front obtained by NS-MFO is shown in Fig. 15, where it justifies the results.

6.6. Spring design problem

The purpose of this problem is to minimize stress and volume simultaneously. The constraints are imposed on the minimum deflection, shear stress, surge frequency, limits on outside diameter and on design variables. The design variables are the wire diameter (d), the mean coil diameter (D), and the number of active coils (N). The schematic view of the spring is shown in Fig. 16. This problem is special because all the design variables possess different characteristics. The number of turns can only take integer values where as diameter of the wire is standardized and it has to be selected from the set of available diameters. The mean coil diameter can be considered as a continuous variable. So, this problem is mixed-integer-discrete problem. The problem can be stated as follows:

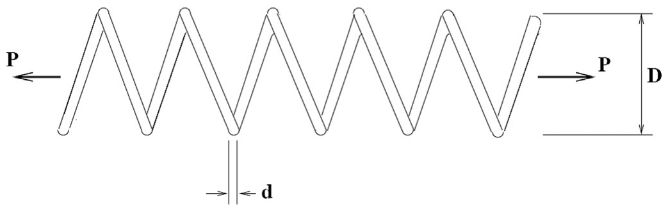

Fig. 16. Spring.

$$Minimize\ f_1(x) = \left(0.253.14^2 x_2^2 x_3 (x_1+2)\right)$$

$$Minimize\ f_2(x) = 8KP_{\_max}\ \frac{x_3}{3.14x_2^3}$$

$$g_1 = 1.05x_2(x_1+2) + \frac{P_{max}}{k} - l_{max}$$

$$g_2 = dmin - x_2$$

$$g_3 = x_2 + x_3 - D_{max}$$

$$g_4 = 3 - C$$

$$g_5 = del_p - del_{pm}$$

$$g_6 = del_w - \frac{P_{max} - P}{k}$$

$$g_7 = 8KP_{max}\frac{x_3}{3.14x_2^3} - S$$

$$g_8 = \left(0.253.14^2 x_2^2 x_3 (x_1+2)\right) - V_{max}$$

Where,

$P = 300$

$D_{\_max} = 3$

$V_{\_max} = 30$

$P_{\_max} = 1000$

$del_w = 1.25$

$l_{\_max} = 14$

$del_{pm} = 6$

$dmin = 0.2$

$S = 189000$

$G = 11500000$

$$C = \frac{x_3}{x_2}$$

$$k = G\frac{x_2^4}{8x_1x_3^3}$$

$$del_p = \frac{P}{k}$$

$$K = \frac{4C-1}{4C-4} + \left(0.615\frac{x_2}{x_3}\right)$$

$$\forall\ g \leq 0$$

$1 \leq x_1 \leq 32, \quad 1 \leq x_3 \leq 30, \quad x_1 \in I$

Table 12
Result (extreme Pareto solutions) for spring problem.

Algorithms		f_1	f_2
NSGA-II	$f_1 \to min$	2.690	187,053
	$f_2 \to min$	24.189	61,949
MOWCA	$f_1 \to min$	2.668	188,448
	$f_2 \to min$	26.93	58,752
NS-MFO	$f_1 \to min$	2.6559	188,088
	$f_2 \to min$	27.96	56,715

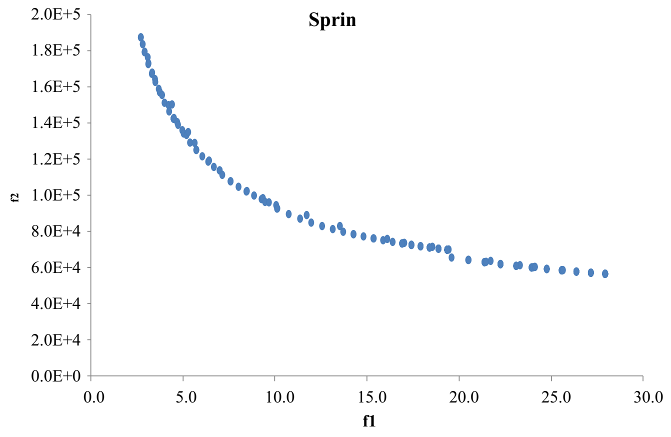

Fig. 17. Pareto front for spring problem.

$x_2 \in [0.009,\ 0.0095,\ 0.0104,\ 0.0118,\ 0.0128,\ 0.0132,\ 0.014,\ 0.015,\ 0.0162,\ 0.0173,\ 0.018,\ 0.020,\ 0.023,\ 0.025,\ 0.028,\ 0.032,\ 0.035,\ 0.041,\ 0.047,\ 0.054,\ 0.063,\ 0.072,\ 0.080,\ 0.092,\ 0.105,\ 0.120,\ 0.135,\ 0.148,\ 0.162,\ 0.177,\ 0.192,\ 0.207,\ 0.225,\ 0.244,\ 0.263,\ 0.283,\ 0.307,\ 0.331,\ 0.362,\ 0.394, 0.4375,\ 0.5]$

This value for GD, S or Δ is not available in the literature for this problem so the results are compared with the extreme Pareto solutions. The extreme Pareto solutions are compared with NSGA-II and MOWCA. The results are presented in the Table 12. It can be observed from the results that the extreme points obtained by NS-MFO are better compared to NSGA-II and MOWCA (Sadollah et al., 2015). The Pareto front is shown in Fig. 17, which justifies the results. The Pareto front of this problem is discontinuous and of overlapping nature. This occurs because each discrete value for *d* possesses certain portion of the Pareto front, which can be observed from the Pareto front for different discrete values of d.

7. Conclusions

Multi-objective version for moth-flame optimization algorithm (MFO) algorithm, called NS-MFO, is proposed in this work. The proposed method works on the concept of non-dominated sorting and crowding distance. The proposed algorithm is experimented on numerical benchmark problems with different characteristics of Pareto front and challenging benchmark problems of CEC 2009. The proposed method is also applied to different engineering design problems with distinctive characteristics. Friedman rank test is performed among 17 algorithms for unconstrained and 9 algorithms for constrained benchmark problems of CEC 2009, which indicate the competitive performance of NS-MFO to the comparative algorithms. The proposed method is also compared with different multi-objective version of GA, DE and WCA for engineering problems. The results indicate that the proposed method is better or at par with the existing algorithms.

Acknowledgments

The authors are thankful to reviewers for considerations, constructive reviews, and beneficial suggestions of the manuscript. These comments have helped us in improving the attractiveness, readability, and quality of the manuscript.

The research of the 2nd author is supported in part by the Natural Sciences and Engineering Research Council of Canada (NSERC) (Grant Number RGPIN-2015-05522). The postdoctoral fellowship of the 1st author is supported by NSERC.

References

Ahrari, A., Atai, A.A., 2010. Grenade explosion method—a novel tool for optimization of multimodal functions. Appl. Soft Comput. 10 (4), 1132–1140.
Akbari, R., Hedayatzadeh, R., Ziarati, K., Hassanizadeh, B., 2012a. A multi-objective artificial bee colony algorithm. Swarm Evolut. Comput. 2, 39–52.
Akbari, R., Hedayatzadeh, R., Ziarati, K., Hassanizadeh, B., 2012b. A multi-objective artificial bee colony algorithm. Swarm Evolut. Comput. 2, 39–52.
Allam, D., Yousri, D.A., Eteiba, M.B., 2016. Parameters extraction of the three diode model for the multi-crystalline solar cell/module using Moth-Flame Optimization Algorithm. Energy Convers. Manag. 123, 535–548.
Angus, D., Woodward, C., 2009. Multiple objective ant colony optimization. Swarm Intell. 3 (1), 69–85.
Aydin, I., Karakose, M., Akin, E., 2011. A multi-objective artificial immune algorithm for parameter optimization in support vector machine. Appl. Soft Comput. 11 (1), 120–129.
Coello, C.A.C., Van Veldhuizen, D.A., Lamont, G.B., 2002. Evolutionary Algorithms for Solving Multi-objective Problems 242. Kluwer Academic, New York.
Deb, K., 2001. Multi-objective Optimization using Evolutionary Algorithms 16. John Wiley & Sons.
Dorigo, M., 1992. Optimization, Learning and Natural Algorithms (Ph.D. thesis). Politecnico di Milano, Italy.
Eskandar, H., Sadollah, A., Bahreininejad, A., Hamdi, M., 2012. Water cycle algorithm–a novel metaheuristic optimization method for solving constrained engineering optimization problems. Comput. Struct. 110, 151–166.
Eusuff, M.M., Lansey, K.E., 2003. Optimization of water distribution network design using the shuffled frog leaping algorithm. J. Water Resour. Plan. Manag. 129 (3), 210–225.
Farmer, J.D., Packard, N.H., Perelson, A.S., 1986. The immune system, adaptation, and machine learning. Phys. D: Nonlinear Phenom. 22 (1), 187–204.
Gong, M., Jiao, L., Du, H., Bo, L., 2008. Multiobjective immune algorithm with nondominated neighbor-based selection. Evolut. Comput. 16 (2), 225–255.
Holland, J.H., 1975. Adaption in natural and artificial systems. In: Proceedings of the 1st International Conference Genetic Aglorithms, Ann Arbor MI: The University of Michigan Press, pp. 93–100.
Jamuna, K., Swarup, K.S., 2012. Multi-objective biogeography based optimization for optimal PMU placement. Appl. Soft Comput. 12 (5), 1503–1510.
Jangir, N., Pandya, M.H., Trivedi, I.N., Bhesdadiya, R.H., Jangir, P., Kumar, A., 2016. March). Moth-Flame Optimization algorithm for solving real challenging constrained engineering optimization problems. In: 2016 IEEE Students' Conference on Electrical, Electronics and Computer Science (SCEECS), IEEE, pp. 1–5.
Kaveh, A., 2014. Charged system search algorithm. In: Advances in Metaheuristic Algorithms for Optimal Design of Structures, Springer International Publishing, pp. 41–85.
Kennedy, V., Eberhart, R., 1995. Particle swarm optimization. In: Proceedings of the IEEE International Conference on Neural Networks, 1942–48.
Khalilpourazari, S., Pasandideh, S.H.R., 2017. Multi-item EOQ model with nonlinear unit holding cost and partial backordering: moth-flame optimization algorithm. J. Ind. Prod. Eng. 34 (1), 42–51.
Krishnanand, K.R., Panigrahi, B.K., Rout, P.K., Mohapatra, A., 2011. Application of multi-objective teaching-learning-based algorithm to an economic load dispatch problem with incommensurable objectives. In: Swarm, Evolutionary, and Memetic Computing, Springer Berlin Heidelberg, pp. 697–705.
Laumanns, M., Thiele, L., Zitzler, E., 2006. An efficient, adaptive parameter variation scheme for metaheuristics based on the epsilon-constraint method. Eur. J. Oper. Res. 169 (3), 932–942.
Li, H., Zhang, Q., 2009. Multiobjective optimization problems with complicated Pareto sets, MOEA/D and NSGA-II. Evolut. Comput. IEEE Trans. 13 (2), 284–302.
Li, X., Zhang, J., Yin, M., 2014. Animal migration optimization: an optimization algorithm inspired by animal migration behavior. Neural Comput. Appl. 24 (7–8), 1867–1877.
Mavrotas, G., 2009. Effective implementation of the ε-constraint method in multi-objective mathematical programming problems. Appl. Math. Comput. 213 (2), 455–465.

Merrikh-Bayat, F., 2015. The runner-root algorithm: a metaheuristic for solving unimodal and multimodal optimization problems inspired by runners and roots of plants in nature. Appl. Soft Comput. 33, 292–303.
Mirjalili, S., 2015. Moth-flame optimization algorithm: a novel nature-inspired heuristic paradigm. Knowl. Based Syst. 89, 228–249.
Mirjalili, S., Mirjalili, S.M., Lewis, A., 2014. Grey wolf optimizer. Adv. Eng. Softw. 69, 46–61.
Mondal, S., Bhattacharya, A., nee Dey, S.H., 2013. Multi-objective economic emission load dispatch solution using gravitational search algorithm and considering wind power penetration. Int. J. Electr. Power Energy Syst. 44 (1), 282–292.
Moslehi, G., Mahnam, M., 2011. A Pareto approach to multi-objective flexible job-shop scheduling problem using particle swarm optimization and local search. Int. J. Prod. Econ. 129 (1), 14–22.
Nikoofard, A.H., Hajimirsadeghi, H., Rahimi-Kian, A., Lucas, C., 2012. Multiobjective invasive weed optimization: application to analysis of Pareto improvement models in electricity markets. Appl. Soft Comput. 12 (1), 100–112.
Omkar, S.N., Senthilnath, J., Khandelwal, R., Naik, G.N., Gopalakrishnan, S., 2011. Artificial Bee Colony (ABC) for multi-objective design optimization of composite structures. Appl. Soft Comput. 11 (1), 489–499.
Passino, K.M., 2002. Biomimicry of bacterial foraging for distributed optimization and control. Control Syst. IEEE 22 (3), 52–67.
Patel, V., Savsani, V., 2014a. Optimization of a plate-fin heat exchanger design through an improved multi-objective teaching-learning based optimization (MO-ITLBO) algorithm. Chem. Eng. Res. Des. 92 (11), 2371–2382.
Patel, V.K., Savsani, V.J., 2014b. A multi-objective improved teaching–learning based optimization algorithm (MO-ITLBO). Inf. Sci.
Patel, V.K., Savsani, V.J., 2015. Heat transfer search (HTS): a novel optimization algorithm. Inf. Sci. 324, 217–246.
Patel, V.K., Savsani, V.J., 2016. A multi-objective improved teaching–learning based optimization algorithm (MO-ITLBO). Inf. Sci. 357, 182–200.
Rao, R.V., Savsani, V.J., Vakharia, D.P., 2011. Teaching–learning-based optimization: a novel method for constrained mechanical design optimization problems. Comput. Aided Des. 43 (3), 303–315.
Rao, R.V., Savsani, V.J., Vakharia, D.P., 2012. Teaching–learning-based optimization: an optimization method for continuous non-linear large scale problems. Inf. Sci. 183 (1), 1–15.
Rashedi, E., Nezamabadi-Pour, H., Saryazdi, S., 2009. GSA: a gravitational search algorithm. Inf. Sci. 179 (13), 2232–2248.
Roy, P.K., Ghoshal, S.P., Thakur, S.S., 2010. Biogeography based optimization for multi-constraint optimal power flow with emission and non-smooth cost function. Expert Syst. Appl. 37 (12), 8221–8228.
Sadollah, A., Eskandar, H., Kim, J.H., 2015. Water cycle algorithm for solving constrained multi-objective optimization problems. Appl. Soft Comput. 27, 279–298.
Sadollah, A., Bahreininejad, A., Eskandar, H., Hamdi, M., 2013. Mine blast algorithm: a new population based algorithm for solving constrained engineering optimization problems. Appl. Soft Comput. 13 (5), 2592–2612.
Savsani, P., Savsani, V., 2016. Passing vehicle search (PVS): a novel metaheuristic algorithm. Appl. Math. Model. 40, 3951–3978.
Savsani, P., Jhala, R.L., Savsani, V., 2014. Effect of hybridizing biogeography-based optimization (BBO) technique with artificial immune algorithm (AIA) and ant colony optimization (ACO). Appl. Soft Comput. 21, 542–553.
Schaffer J.D., 1985. Multiple Objective Optimization with Vector Evaluated Genetic Algorithms.
Shareef, H., Ibrahim, A.A., Mutlag, A.H., 2015. Lightning search algorithm. Appl. Soft Comput. 36, 315–333.
Simon, D., 2008. Biogeography-based optimization. Evolut. Comput. IEEE Trans. 12 (6), 702–713.
Storn, R., Price, K., 1997. Differential evolution–a simple and efficient heuristic for global optimization over continuous spaces. J. Glob. Optim. 11 (4), 341–359.
Tripathi, P.K., Bandyopadhyay, S., Pal, S.K., 2007. Multi-objective particle swarm optimization with time variant inertia and acceleration coefficients. Inf. Sci. 177 (22), 5033–5049.
Wang, G.G., 2016. Moth search algorithm: a bio-inspired metaheuristic algorithm for global optimization problems. Memetic Comput., 1–14.
Wang, G.G., Deb, S., Coelho, L.D.S., 2015a. Earthworm optimization algorithm: a bio-inspired metaheuristic algorithm for global optimization problems. Int. J. Bio-Inspired Comput.
Wang, G.G., Deb, S., Cui, Z., 2015b. Monarch butterfly optimization. Neural Comput. Appl., 1–20.
Wang, G.G., Deb, S., Gao, X.Z., Coelho, L.D.S., 2016. A new metaheuristic optimisation algorithm motivated by elephant herding behaviour. Int. J. Bio-Inspired Comput. 8 (6), 394–409.
Wang, Y., Yang, Y., 2009. Particle swarm optimization with preference order ranking for multi-objective optimization. Inf. Sci. 179 (12), 1944–1959.
Yagmahan, B., Yenisey, M.M., 2008. Ant colony optimization for multi-objective flow shop scheduling problem. Comput. Ind. Eng. 54 (3), 411–420.
Yang, X.S., 2009. Firefly algorithms for multimodal optimization. In: Stochastic Algorithms: Foundations and Applications, Springer Berlin Heidelberg, pp. 169–178.
Yang, X.S., 2010. A new metaheuristic bat-inspired algorithm. In: Nature inspired cooperative strategies for optimization (NICSO 2010), Springer Berlin Heidelberg, pp. 65–74.
Yang, X.S., 2011. Bat algorithm for multi-objective optimisation. Int. J. Bio-Inspired Comput. 3 (5), 267–274.
Yang, X.S., 2013. Multiobjective firefly algorithm for continuous optimization. Eng. Comput. 29 (2), 175–184.
Yang, X.S., Deb, S., 2010. Engineering optimisation by cuckoo search. Int. J. Math. Model. Numer. Optim. 1 (4), 330–343.
Yang, X.S., Deb, S., 2013. Multiobjective cuckoo search for design optimization. Comput. Oper. Res. 40 (6), 1616–1624.
Yazdani, M., Jolai, F., 2015. Lion optimization algorithm (LOA): a nature-inspired metaheuristic algorithm. J. Comput. Des. Eng.
Zhang, H., Zhu, Y., Zou, W., Yan, X., 2012. A hybrid multi-objective artificial bee colony algorithm for burdening optimization of copper strip production. Appl. Math. Model. 36 (6), 2578–2591.
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. University of Essex, Colchester, UK and Nanyang technological University, Singapore, Special Session on Performance Assessment of Multi-objective Optimization Algorithms, Technical Report, 264.
Zitzler, E., Deb, K., Thiele, L., 2000. Comparison of multiobjective evolutionary algorithms: empirical results. Evolut. Comput. 8 (2), 173–195.