A Multi-Objective Gravitational Search Algorithm Based on Non-Dominated Sorting

Hadi Nobahari, Sharif University of Technology, Iran

Mahdi Nikusokhan, Sharif University of Technology, Iran

Patrick Siarry, Université Paris-Est Créteil (UPEC), France

ABSTRACT

This paper proposes an extension of the Gravitational Search Algorithm (GSA) to multi-objective optimization problems. The new algorithm, called Non-dominated Sorting GSA (NSGSA), utilizes the non-dominated sorting concept to update the gravitational acceleration of the particles. An external archive is also used to store the Pareto optimal solutions and to provide some elitism. It also guides the search toward the non-crowding and the extreme regions of the Pareto front. A new criterion is proposed to update the external archive and two new mutation operators are also proposed to promote the diversity within the swarm. Numerical results show that NSGSA can obtain comparable and even better performances as compared to the previous multi-objective variant of GSA and some other multi-objective optimization algorithms.

Keywords: *Gravitational Search Algorithm, Multi-Objective Optimization, Non-Dominated Sorting, Reordering Mutation, Sign Mutation*

INTRODUCTION

Over the last decades, several meta-heuristics have been developed to solve complex single and multi-objective optimization problems. Many real-world problems involve simultaneous optimization of several competing objectives. In these problems, there is no single optimal solution, but rather a set of non-dominated solutions, also called Pareto-optimal solutions. The use of meta-heuristics to solve multi-objective optimization problems is growing fast, because they can handle problems with concave and disconnected Pareto fronts.

There are many meta-heuristics such as Simulated Annealing (SA) (Kirkpatrick et al., 1983), Tabu Search (TS) (Glover, 1989, 1990), Evolutionary Algorithms (EAs) (Tang et al., 1996), Ant Colony Optimization (ACO) (Dorigo et al., 1996), Particle Swarm Optimization (PSO) (Kennedy & Eberhart, 1995), Gravitational Search Algorithm (GSA) (Rashedi et al., 2009) and so on. These algorithms are able to solve different optimization problems. However, there is no specific algorithm able to find the best solutions of all problems in finite

DOI: 10.4018/jsir.2012070103

iterations and some algorithms show better performances for particular problems than the others. Hence, searching for new heuristic optimization algorithms is an open problem.

Although both single and multi-agent metaheuristics have their own multi-objective variants, the multi-agent meta-heuristics such as EAs, ACO, and PSO have shown better potential to solve multi-objective optimization problems, because in a multi-objective problem, a population of solutions is going to be generated, in a single run. The multi-objective variants of the single agent meta-heuristics such as SA and TS are often working based on the definition of an aggregation function in their structure (Ulungu et al., 1999; Hansen, 1997). The use of aggregation functions may reduce the search area and provide only a subset of the Pareto front.

GSA is a new multi-agent optimization algorithm, inspired from the general gravitational law (Rashedi et al., 2009). The algorithm is based on the movement of some particles under the effect of the gravitational forces, applied by the others. Hassanzadeh and Rohani (2010) have proposed the first multi-objective variant of GSA, called Multi-Objective GSA (MOGSA). MOGSA uses an external archive to store the non-dominated solutions and uses the same idea as in Simple Multi-Objective PSO (SMOPSO), proposed by Cagnina et al. (2005), to update the external archive. For this purpose, the interval of each objective function is divided into equal divisions and consequently the entire performance space is divided into hyper-rectangles. When the number of archived members exceeds the maximum archive length, one member of the most crowded hyper-rectangle is randomly chosen and removed.

The main problem in proposing a multi-objective variant for GSA is updating the mass of particles based on the value of the multiple objectives. In MOGSA, the mass of all moving particles is set to one and the mass of archived particles is updated based on the distance from the nearest neighbor, within the objective space. However, no equation has been presented that relates the mass value to the distance value. This technique distributes the archived elements uniformly, similar to the niching technique. After calculating the mass of the moving particles, they move to the new positions by the forces applied from the archived members. In MOGSA, only the archived particles apply gravitational forces to the moving ones and after each movement, the well-known uniform mutation is applied to the new positions.

In this paper, the authors have proposed a new multi-objective variant of GSA, called NSGSA. In single-objective GSA, proposed by Rashedi et al. (2009), the mass of each particle is taken to be proportional to its fitness. As mentioned, the main problem to propose a multi-objective GSA is to establish a relationship between the mass of each particle and its multiple objectives. In this paper, the non-dominated sorting concept, proposed in Non-dominated Sorting GA (NSGA) (Srinivas & Deb, 1994), is used to divide the particles to several layers within the performance space. In this way, the mass of each particle will depend to the rank of the layer it belongs to. NSGSA utilizes a limited length external archive to store the last found non-dominated solutions. To provide some elitism, specific members of the external archive are added to the list of moving particles and contribute in applying gravitational forces to the others. A new criterion is introduced to update the external archive, based on a new defined spread indicator. The mutation (turbulence) phenomenon is also modeled in NSGSA, not considered in the original GSA. In this regard, two new mutation operators, called sign and reordering mutations, are also proposed.

The paper is organized as follows: At first, GSA is described in detail. Thereafter, the new multi-objective variant of GSA, called NSGSA, is introduced. Next, numerical results are presented. Then, the results of NSGSA are compared with those of MOGSA, SMOPSO, and NSGA-II (Deb et al., 2002) and their sensitivity to the value of tuned parameters is also investigated. Finally, the conclusions are outlined.

GRAVITATIONAL SEARCH ALGORITHM

GSA is a new optimization algorithm, proposed by Rashedi et al. (2009). This algorithm has been inspired from the mass interactions based on the general gravitational law. In the proposed algorithm, the search agents are a collection of masses, interact with each other based on the Newtonian laws on gravity and motion. Also, it uses some stochastic operators to increase diversity and the probability of finding the global optimum. In the following, the formulation of GSA is presented in short. The mass of each particle is calculated according to its fitness value, as follows:

$$m_i(t) = \frac{\mathrm{fit}_i(t) - \mathrm{worst}(t)}{\mathrm{best}(t) - \mathrm{worst}(t)} \quad i = 1,2,\ldots,\mathrm{n}_{\mathrm{mass}}$$
$$M_i(t) = \frac{m_i(t)}{\sum_{j=1}^{\mathrm{n}_{\mathrm{mass}}} m_j(t)} \qquad 0 \le M_i(t) < 1 \tag{1}$$

where t is time (iteration), fit$_i$ presents fitness value of the i-th particle, M$_i$ is the normalized mass, n$_{\mathrm{mass}}$ is the number of particles, and worst(t) and best(t) are defined for a minimization problem as follows:

$$\mathrm{worst}(t) = \max_{i=1}^{\mathrm{n}_{\mathrm{mass}}}\{\mathrm{fit}_i(t)\} \quad , \quad \mathrm{best}(t) = \min_{i=1}^{\mathrm{n}_{\mathrm{mass}}}\{\mathrm{fit}_i(t)\} \tag{2}$$

The vector of force, applied by mass j to mass i, $\mathbf{f}_{ij}(t)$, is calculated as follows:

$$\mathbf{f}_{ij}(t) = G(t)\frac{M_i(t)\,M_j(t)}{R_{ij}(t) + \varepsilon}(\mathbf{x}_j - \mathbf{x}_i) \tag{3}$$

where x$_i$ is the position vector of the i-th agent, ε is a small threshold, and G(t)= G(G$_0$, t) is the gravitational coefficient, initialized at the beginning of the algorithm and is reduced with time to control the search accuracy, R$_{ij}$(t) is the Euclidian distance between two agents i and j, defined as follows:

$$R_{ij}(t) = \left\|\mathbf{x}_i(t) - \mathbf{x}_j(t)\right\|_2 \tag{4}$$

Using the second Newton's law of motion, the gravitational acceleration of the i-th agent due to the j-th one at time t is given as follows:

$$\mathbf{a}_{ij}(t) = G(t)\frac{M_j(t)}{R_{ij}(t) + \varepsilon}(\mathbf{x}_j - \mathbf{x}_i) \tag{5}$$

Actually, the gravitational force between two particles is inversely proportional to the square of the relative distance (Holliday et al., 1993), and the Newtonian gravitational acceleration can be stated as follows:

$$\mathbf{a}_{ij}(t) = G(t)\frac{M_j(t)}{R_{ij}^3(t) + \varepsilon}(\mathbf{x}_j - \mathbf{x}_i) \tag{6}$$

Rashedi et al. (2009) used Equation (5) instead of Equation (6), since it provides better results, as they have reported from their experiments. Therefore, the total gravitational acceleration of the i-th agent, is calculated as follows:

$$\mathbf{a}_i(t) = G(t)\sum_{j=1}^{\mathrm{n}_{\mathrm{mass}}} rand_j \frac{M_j(t)}{R_{ij}(t) + \varepsilon}(\mathbf{x}_j - \mathbf{x}_i) \tag{7}$$

where rand$_j$ is a uniform random number within the interval [0,1], considered to add some stochastic behavior to the acceleration. The velocity and position of the agents are updated as follows:

$$\begin{aligned}\mathbf{v}_i(t+1) &= rand_i\,\mathbf{v}_i(t) + \mathbf{a}_i(t)\\ \mathbf{x}_i(t+1) &= \mathbf{x}_i(t) + \mathbf{v}_i(t+1)\end{aligned} \tag{8}$$

where $rand_i$ is another uniform random number in the interval [0,1]. To prevent the particles go out of the search space, the positions are bounded as follows:

$$x_i^d = \begin{cases} x_l^d & x_i^d < x_l^d \\ x_u^d & x_i^d > x_u^d \end{cases} \tag{9}$$

where d=1,…, n shows the dimension of the search space and x_l^d and x_u^d are lower and upper bounds of the search interval in dimension d, respectively.

To make a good compromise between exploration and exploitation, Rashedi et al. (2009) proposed to reduce the number of attractive agents in Equation (7) over time. Hence, only a set of best agents, i.e., the agents with bigger mass, apply gravitational forces to the others. However, this policy may reduce the exploration power and increase the exploitation capability. In order to avoid being trapped in local optima, the algorithm must perform more exploration at the beginning. Over time (iteration), exploration must be gradually faded out and exploitation must be faded in. Therefore, in GSA, only the $k_{best}(t)$ of the moving agents will attract the others. $k_{best}(t)$ is a decreasing function of time, with the initial value $k_0=n_{mass}$. It means that at the beginning, all agents apply gravitational forces, and as time passes, $k_{best}(t)$ is decreased linearly. At the end, there will be just one agent applying force to the others. Therefore, Equation (7) could be modified as follows:

$$\mathbf{a}_i(t) = G(t) \sum_{j \in K_{\text{best}}(t)} rand_j \frac{M_j(t)}{R_{ij}(t) + \varepsilon} (\mathbf{x}_j - \mathbf{x}_i) \tag{10}$$

where $K_{best}(t)$ is the set of best $k_{best}(t)$ agents, found so far.

Sarafrazi et al. (2011) proposed a new operator, called disruption to improve the exploration and exploitation of GSA. The authors have claimed that this operator has been inspired from a natural phenomenon in astrophysics. In this method, all solutions except the best one, are disrupted. The disruption is occurred, if the ratio of the distance between mass i and its nearest neighbor, *j*, to its distance from the best solution is smaller than a specific threshold as follows:

$$\frac{R_{ij}}{R_{i,\text{best}}} < \text{C} \tag{11}$$

Where R_{ij} and $R_{i,best}$ are Euclidian distances between solutions i and j and between solution i and the best solution, respectively, and C is a specific threshold. The position of every mass that satisfies the condition will be changed as follows:

$$x_i(new) = x_i(old)\,D$$
$$D = \begin{cases} R_{ij}(-0.5 + rand) & R_{i,best} > 1 \\ 1 + \rho(-0.5 + rand) & otherwise \end{cases} \tag{12}$$

Again, $rand$ is a uniform random number within the interval [0,1] and ρ is a small number. Li et al. (2011) have proposed an improved variant of GSA, called Improved GSA (IGSA). This algorithm is a combination of GSA and PSO. In IGSA, the velocity is updated as follows:

$$\begin{aligned} &\mathbf{v}_i(t+1) \\ &= rand_{i_1}\ \mathbf{v}_i(t) + \mathbf{a}_i(t) \\ &+ \text{c}_1\ rand_{i_2} (\mathbf{p}_{\text{best},i} - \mathbf{x}_i) \\ &+ \text{c}_2\ rand_{i_3} (\mathbf{g}_{\text{best}} - \mathbf{x}_i) \end{aligned} \tag{13}$$

Where $rand_{i_1}$, $rand_{i_2}$ and $rand_{i_3}$ are uniform random numbers within the interval [0,1], c_1 and c_2 are constants, $p_{best,i}$ is the best previous position of the ith particle and g_{best} is the best previous position of all particles.

NON-DOMINATED SORTING GRAVITATIONAL SEARCH ALGORITHM

In this section the new multi-objective variant of GSA, called NSGSA, is introduced. The new algorithm utilizes the concept of Pareto optimality condition and non-dominance. A high level description of NSGSA is presented

Figure 1. Pseudo-code of NSGSA

```
Initialization()
t=1
While t<tmax
    Evaluate Fitness of each particle
    Update external archive()
    Non-dominated sorting()
    Update the list of moving particles()
    Update the mass of moving particles()
    Update particles acceleration()
    Update particles velocity()
    Update and Mutate particles position()
    t=t+1
EndWhile
Report Pareto optimal solutions stored in archive
```

in the following subsection, details of which are provided in later subsections.

General Setting Out of the Algorithm

Figure 1 shows the general iterative steps of NSGSA in the form of a pseudo code. The main loop starts after initializing the particle positions and velocities. During any iteration, the fitness of particles is evaluated. An external archive of the Pareto optimal solutions with limited size is generated in the first iteration. In later iterations, the updated particles are compared with the members of the external archive and the archive is updated based on the Pareto optimality condition. When the external archive is completed and a new member is going to be added, a spread indicator is utilized to decide which solution to be replaced. The moving particles are then ranked in layers using the non-dominated sorting algorithm (Srinivas & Deb, 1994). To add some elitism to the algorithm and to promote the uniform distribution of the solutions, some specific members of the external archive are added to the list of moving particles and instead the same numbers of the moving particles are discarded from the worst layer(s).

As in single objective GSA the mass of each particle is a function of time and depends on its fitness. But, in a multi-objective problem, there are more objectives than one. NSGSA utilizes the non-dominated sorting algorithm to assign each particle a rank, regarding the layer it belongs to. Then, rank of each particle is considered as its fitness and the mass is calculated using Equation (1).

The accelerations, exerted to the particles, are calculated according to the varying gravitational constant, using the same relations, proposed in GSA (Rashedi et al., 2009). Then, the velocity and position of particles are updated according to the kinematic relations. The position of particles are also mutated to increase the diversity of the algorithm and to prevent premature convergences. The mutation (or turbulence) strength depends on the velocity of particles. As the gravitational constant and consequently the velocity of particles are decreased over time, the mutation strength is decreased, too. Therefore, exploration is decreased through the iterations while at the same time exploitation increases and helps to find the final solutions accurately. In the following subsections, the steps, bolded in Figure 1 are discussed in detail.

Initialization

The new algorithm has eight control parameters, defined in the next subsections, which must be set before the execution of the algorithm. An empty external archive is generated. Initial position of particles is set randomly within the search intervals and the initial velocities are set to zero for all particles.

Updating the External Archive

The external archive is updated based on the Pareto dominance criterion. If a member of the swarm is dominated by any member of the external archive, it will not be inserted to the archive. In contrast, if it is non-dominated with respect to all members of the archive or it dominates some members of the archive, it will be inserted into the archive and the dominated member(s) will be removed. If the number of archived members exceeds the maximum archive length, the most crowded area must be determined to remove a member from that area. A spread indicator is introduced in the paper to control the length of external archive. This indicator is based on the spread of the points within the Pareto front. Here, we are interested in getting a set of solutions that spans the entire Pareto optimal region as uniformly as possible. For this purpose, a crowding distance measure, $d_{c,i}$, similar to that proposed by Deb et al. (2002), is defined as follows:

$$d_{c,i} = \sqrt{\sum_{j=1}^{m}\left(d_{c,i}^{j}\right)^2} \quad (i=1,..,n_{archive}) \tag{14}$$

Where

$$d_{c,i}^{j} = fit_{i+1}^{j} - fit_{i-1}^{j} \quad (i=1,..,n_{archive}, j=1,..,m) \tag{15}$$

and fit_{i+1}^{j} is the j-th fitness of the point i+1 when the members of the archive are sorted versus the j-th objective function. It is desired to have nearly equal crowding distances. To have nearly equal $d_{c,i}$, the deviation of these distances from their average, $\bar{d}_c$, should be minimized. For this purpose a spread indicator, δ, is defined as follows:

$$\delta = \sum_{\substack{i=1 \\ i \notin E}}^{n_{archive}} \left| d_{c,i} - \bar{d}_c \right| \Big/ \left[n_{archive} - m \right] \bar{d}_c \tag{16}$$

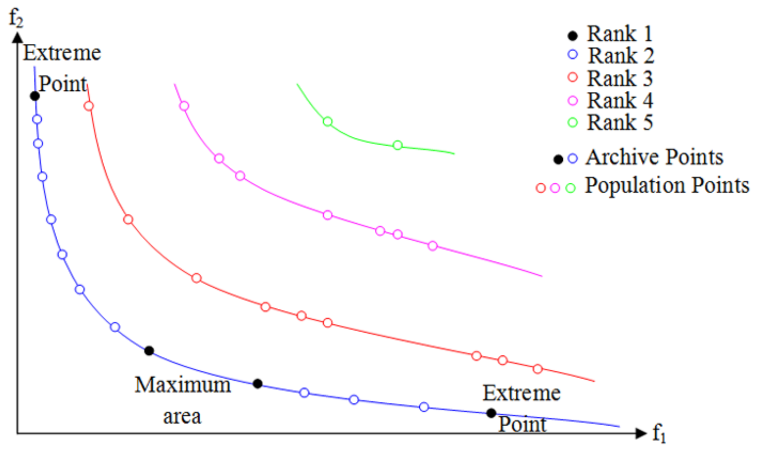
where $n_{archive}$ is the archive length and E is the set of extreme points, the size of which is taken equal to m. Deb et al. (2002) use a similar equation for measuring the performance of the algorithms, called as spread metric. To obtain a uniform spread of Pareto archive, δ must be decreased. In order to decrease δ, the difference between $\max\{d_{c,i}\}$ and $\min\{d_{c,i}\}$ must be decreased ($i=1,\ldots, n_{archive}$). In other words, the maximum $d_{c,i}$ must be decreased or the minimum $d_{c,i}$ must be increased. Inserting a new member to the archive decreases the maximum $d_{c,i}$ or at least does not change it, but increasing the minimum value is only possible through removing a suitable member. For this purpose, the two nearest neighbors in the objective space, are determined and named as members 1 and 2. These two members are removed alternatively to calculate δ_1 and δ_2 corresponding to the removal of members 1 and 2, respectively. The member, the presence of which causes bigger δ is then removed, unless it extends the Pareto front, the case of which the other member should be removed.

Updating the List of Moving Particles

During any iteration, some members of the external archive should be inserted to the list of moving particles for several reasons: First, insertion of such members adds some elitism to the algorithm. Next, if the extreme points of the external archive are also inserted, then the algorithm will hopefully be able to extend the Pareto front. Finally, insertion of the members, located in the least crowded area, can repair the search gaps within the Pareto front.

The list of moving particles for the next flight is updated as follows: m extreme points of the Pareto front (m single-objective optimal solutions of the external archive) are inserted to the list. Then, m points, located in the least

Figure 2. Non-dominated sorting and ranking



crowded area, are inserted. These points are selected using the crowding distance assignment algorithm, proposed by Deb et al. (2002). A percent of the archive members, $P_{elitism}$, is also selected randomly and inserted to the list of moving particles. Finally, the list length is limited to n_{mass} by removing some of the worst particles, starting from the worst layer of the non-dominated sorted particles. It should be noted that the initial velocity of the members, inserted from the external archive to the list of moving particles, must be set to zero.

Updating the Mass of Moving Particles

As in single objective GSA, the mass of each particle is a function of time (iteration). It depends on the fitness of particle and is calculated using Equation (1). In a multi-objective optimization problem, there are multiple objectives and a single fitness function must be defined corresponding to each solution. NSGSA utilizes the ranks of the layers, generated using the non-dominated sorting algorithm to determine the fitness of each particle.

In NSGSA, the m members, imported from the extreme regions of the external archive and the m members, imported from the least crowded area of this archive are ranked as 1, as depicted in Figure 2. The $P_{elitism}$ percent of the archive members, chosen randomly and inserted to the list of moving particles are ranked as 2. The members, remained from the last iteration, get ranks 3 and more, regarding the rank of the layers they belong to. The rank of each particle is then considered as its fitness and its mass is updated using Equation (1).

It should also be noted that the imported particles from the archive usually have bigger mass than the others due to their better fitness. Therefore, according to Equation (10) their acceleration is less than the other particles. Moreover, as mentioned previously, the initial velocity of these new imported particles is set to zero. Therefore, these new particles move more slowly than the others and provide fine search around the Pareto front.

Updating the Acceleration of Particles

NSGSA uses the same equation as GSA to update the acceleration of particles, i.e., Equation (10). The most important parameter of this equation is the gravitational constant, $G(t)$. Rashedi et al. (2009) suggest $G(t)$ as follows:

$$G(t) = G_0 e^{-\frac{\alpha t}{t_{max}}} \tag{17}$$

where α and G_0 are the function parameters. It is difficult to find a suitable constant G_0 for various test functions. In this paper, G_0 is proposed as follows:

$$G_0 = \beta \max_{d \in \{1,..,n\}} \left(\left| x_u^d - x_l^d \right| \right) \tag{18}$$

where β is a parameter of NSGSA. According to Equation (10), the accelerations, exerted to the particles, depend to G(t). High values of G(t) indicates that the particles can move in long steps, suitable for exploration. Similarly, low values of G(t) indicates that the particles can move in short steps, suitable for exploitation. The quick decrease of G(t) in Equation (17) causes a fast decay of the exploration. To improve this problem NSGSA utilizes a linear function of G(t), defined as follows:

$$G(t) = G_0 \times (1 - t / t_{\max}) \tag{19}$$

The Use of Mutation Operator

To the authors' experience, the original GSA suffers from the premature convergence in the case of complex multimodal problems. To solve this problem, a mutation operator is added to the original GSA to decrease the possibility of trapping in local optima. In reality, the close particles may impact to each other and consequently the direction of their movement may change. Therefore, to simulate the irregularities, exist in the movement of the real particles, two new mutation operators have been proposed, called sign and reordering mutation. NSGSA utilizes a combination of these two mutation operators as its mutation (turbulence) operator. The new mutation operators, proposed by the authors, are defined in the following.

Sign Mutation. In sign mutation, to update the position of each particle, the sign of velocity vector changes temporally, with a predefined probability, P_s, as follows:

$$\begin{aligned} & v_i'^d(t+1) = s_i^d \, v_i^d(t+1) \\ & (d = 1,..,n,\ i = 1,..,n_{\text{mass}}) \\ & s_i^d = \begin{cases} -1 & rand < P_s \\ 1 & \text{otherwise} \end{cases} \\ & \mathbf{x}_i(t+1) = \mathbf{x}_i(t) + \mathbf{v}_i'(t+1) \end{aligned} \tag{20}$$

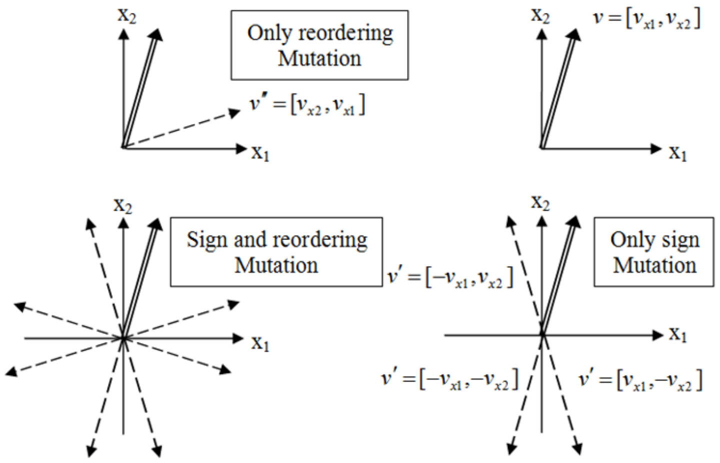
where $v_i'^d(t+1)$ is the mutated velocity by the sign mutation operator and rand is a uniform random number, generated in the interval [0,1]. Ho et al. (2005) propose a similar mutation operator. But, there are differences between these two mutations. The sign mutation, proposed by Ho et al. (2005), changes the sign of $v_i^d(t)$; whereas, the sign mutation, proposed here, mutate the position of the particle, by temporally changing the sign of $v_i^d(t+1)$ to update the position. Then, in the later iteration, the non-mutated velocity, $v_i^d(t+1)$, is used to calculate $v_i^d(t+2)$.

Reordering Mutation. In this mutation, some particles are chosen randomly to be mutated according to reordering mutation probability (P_r). Then, elements of the velocity vector are rearranged randomly. In Figure 3, different possible mutations of a velocity vector are shown for a two dimension space. As can be seen, when the sign and the reordering mutations are applied together, the exploration space is increased with respect to the case when only one of them is applied.

Update and Mutate the Position of Particles

NSGSA utilizes a combination of sign and reordering mutations to update the position of particles. The mutation operator of NSGSA works as follows: the velocity of particle is first mutated by the sign and then by the reordering mutations. Therefore, in NSGSA the position of particles is updated as follows:

Figure 3. Sign and reordering mutation operators and their combination



$$\begin{aligned}
\mathbf{v}_i(t+1) &= w(t) \times \mathbf{v}_i(t) + \mathbf{a}_i(t) \\
\mathbf{v}'_i(t+1) &= \mathbf{Sign\_Mutate}(\mathbf{v}_i(t+1)) \\
\mathbf{v}''_i(t+1) &= \mathbf{Reordering\_Mutate}(\mathbf{v}'_i(t+1)) \\
\mathbf{x}_i(t+1) &= \mathbf{x}_i(t) + \mathbf{v}''_i(t+1)
\end{aligned} \tag{21}$$

where w(t) is the time varying inertia coefficient. In original GSA, this coefficient is taken a random number between 0 and 1. As in PSO, it is better to use a decreasing inertia weight than a random one to have proper exploration and exploitation in the first and the last iterations, respectively. NSGSA utilizes a time varying weighting coefficient as follows:

$$w(t) = \mathrm{w}_0 - (\mathrm{w}_0 - \mathrm{w}_1)\frac{t}{t_{\max}} \tag{22}$$

It should be noted that the mutation operator of NSGSA is applied to the position vector not the velocity vector.

NUMERICAL RESULTS

To evaluate the performance of NSGSA, the optimization results, obtained for a set of standard benchmarks, are compared with the well-known NSGA-II, MOGSA and SMOPSO. Deb et al. (2002) used nine benchmark problems, known as SCH, FON, POL, KUR, ZDT1, ZDT2, ZDT3, ZDT4 and ZDT6, to evaluate the performance of NSGA-II. To compare the results of NSGSA with those of NSGA-II, the same benchmarks are utilized here. Cagnina et al. (2005) and Hassanzadeh and Rohani (2010) have used three benchmarks, known as Viennet3 (MOP5), Deb (MOP6) and Binh2 (MOPC1), to evaluate the performance of MOGSA and SMOPSO, respectively. These benchmarks are also utilized here, to compare the performance of NSGSA with that of MOGSA and SMOPSO.

Some metrics are used to measure the performance of multi-objective algorithms, the examples of which can be found in Cagnina et al. (2005) and Deb et al. (2002). In this study, the two metrics, defined by Deb et al. (2002), are used to compare the performance of NSGSA with NSGA-II. The first metric (γ), measures the convergence of Pareto front, obtained by the algorithm to a known set of true Pareto optimal solutions. The second metric (Δ), called the diversity metric, measures the spread, achieved among the solutions and the extension of the Pareto front. In the same way, to compare the

Table 1. Parameters of NSGSA and their tuned values

Description	Parameters	value
Swarm size	n_{mass}	100
Archive length	$n_{archive}$	100/800
Reordering mutation probability	P_r	0.4
Sign mutation probability	P_s	0.9
Percent of elitism	$P_{elitism}$	0.5
Initial value of inertial coefficient	w_0	0.9
Final value of inertial coefficient	w_1	0.5
Coefficient of search interval	β	2.5

performance of NSGSA with that of MOGSA and SMOPSO, the two metrics, defined by Hassanzadeh and Rohani (2010), are utilized. These two metrics, named as Generational Distance (GD) and Spacing (S) are used to measure the convergence and the spread of Pareto front, respectively.

Parameter Setting

NSGSA has eight control parameters. These parameters and their tuned values are listed in Table 1. To make the results comparable with those of NSGA-II, the number of function evaluations is considered to be 25000, as in Deb et al. (2002). Therefore, the maximum number of iterations, t_{max}, is 250. Also, the length of external archive (narchive) is set to 100. Similarly, to compare results with MOGSA and SMOPSO, the number of function evaluations is considered to be 210000 for MOP5, 60000 for MOP6 and 40000 for MOPC1, as in Cagnina et al. (2005) and Hassanzadeh and Rohani (2010) and the length of external archive is set to 800.

Results and Discussion

The results, presented by Deb et al. (2002), are for real-coded and binary coded variants of NSGA-II, none of which outperforms the other

Table 2. Mean and variance of the convergence metric γ, obtained for NSGSA and both variant of NSGA-II

Algorithm		SCH	FON	POL	KUR	ZDT1	ZDT2	ZDT3	ZDT4	ZDT6
NSGSA	$\bar{\gamma}$	0.003	0.001	0.013	0.017	**0.001**	0.002	0.004	7.251	0.003
	σ_{γ}	0	0	0.000	0	**0.000**	0.001	0.000	3.526	0.002
NSGA-II (real-coed)	$\bar{\gamma}$	0.003	0.002	0.015	0.029	**0.033**	0.072	0.114	0.513	0.296
	σ_{γ}	0	0	0.000	0.000	**0.005**	0.032	0.008	0.118	0.013
NSGA-II (binary-coded)	$\bar{\gamma}$	0.003	0.002	0.017	0.029	**0.001**	0.001	0.043	3.226	7.807
	σ_{γ}	0	0.000	0.000	0.000	**0**	0	0.000	7.307	0.002

Figure 4. Variation of the normalized convergence metric, γ, vs. iteration for a sample run

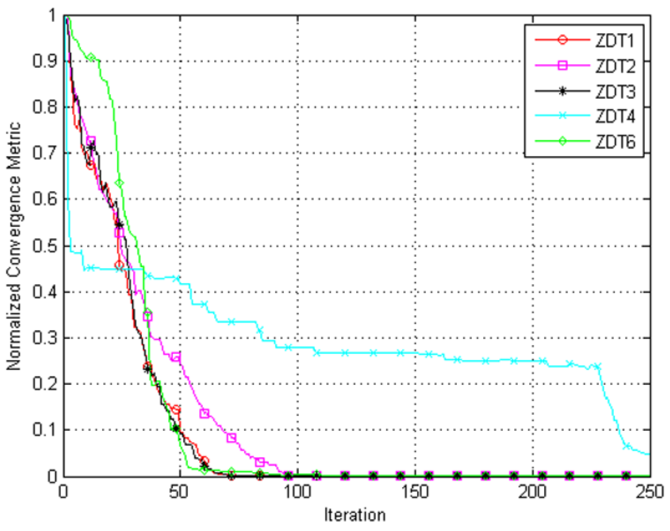


Table 3. Mean and variance of the spread metric Δ, obtained for NSGSA and both variant of NSGA-II

Algorithm		SCH	FON	POL	KUR	ZDT1	ZDT2	ZDT3	ZDT4	ZDT6
NSGSA	$\bar{\Delta}$	0.004	0.005	0.010	0.013	**0.014**	0.050	0.246	0.821	0.487
	σ_{Δ}	0.003	0.003	0.025	0.005	**0.037**	0.080	0.001	0.110	0.005
NSGA-II (real-coed)	$\bar{\Delta}$	0.478	0.378	0.452	0.411	**0.390**	0.431	0.738	0.703	0.668
	σ_{Δ}	0.003	0.000	0.003	0.001	**0.002**	0.005	0.020	0.065	0.010
NSGA-II (binary-coded)	$\bar{\Delta}$	0.449	0.395	0.503	0.442	**0.463**	0.435	0.575	0.479	0.644
	σ_{Δ}	0.002	0.001	0.004	0.001	**0.041**	0.025	0.005	0.009	0.035

in all benchmarks. NSGSA is compared with both variants of NSGA-II. Table 2 compares the mean and variance of the convergence metric, γ, obtained using NSGA-II (real-coded), NSGA-II (binary-coded) and NSGSA. As can be seen, regarding the mean value of γ, NSGSA has better converge in FON, POL, KUR, ZDT1, ZDT3, and ZDT6 problems than the two other algorithms, while in ZDT4, the real-coded NSGA-II, and in ZDT2, the binary-coded variant outperform the others, and in SCH all algorithms have the same convergence rate. Figure 4 shows the variation of γ during a sample run, obtained for NSGSA. To make the results comparable, the curves have been normalized between 0 and 1. Table 3 shows the mean and

Figure 5. Variation of the diversity metric Δ vs. iteration for a sample run

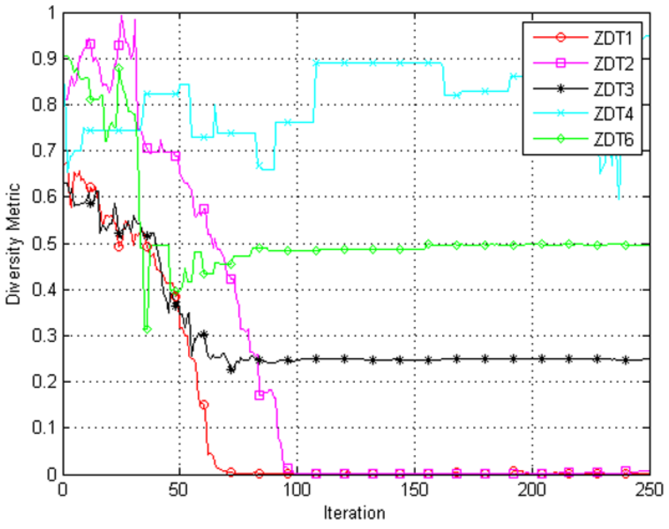


Table 4. Mean and variance of the CPU time obtained for NSGSA

parameter	SCH	FON	POL	KUR	ZDT1	ZDT2	ZDT3	ZDT4	ZDT6
$\overline{T}$ (sec)	16.5	39.1	26.7	32.5	136.2	130.6	129.8	36.7	70.3
σ_T (sec)	0.4	0.5	0.7	0.9	1.1	0.6	0.1	2.3	1.8

variance of the diversity metric, Δ, obtained using the three algorithms. The Pareto front, obtained using NSGSA has better diversity than the two other algorithms in all benchmarks except in ZDT4, regarding the mean value of the diversity metric. In ZDT4, the binary-coded NSGA-II has obtained a little better spread with respect to the others. Figure 5 shows the variation of diversity metric versus iteration for a sample run, obtained for NSGSA. As can be seen, the diversity metric can be deteriorated when a new optimal Pareto solution is found and inserted to the external archive.

All experiments took place on a 2.13 GHz Intel laptop machine. Table 4 shows the mean and variance of the CPU time, shown by $\overline{T}$ and σ_T , respectively.

Table 5. Mean of generational distance metric, obtained for NSGSA, SMOPSO and MOGSA

Algorithm	MOP5	MOP6	MOPC1
NSGSA	1e-4	3e-5	0.001
SMOPSO	0.011	3e-4	0.003
MOGSA	0.001	2e-5	0.004

Table 6. Mean of spacing metric, obtained for NSGSA, SMOPSO and MOGSA

Algorithm	MOP5	MOP6	MOPC1
NSGSA	0.034	5e-4	0.057
SMOPSO	0.396	0.003	0.116
MOGSA	0.181	0.001	0.180

Table 7. Variation of the mean performance metrics vs. the inertia weight

Benchmark		w0=0.9,w1=0.5	w(t) =0.5	w(t) =0.9	w(t) = rand
ZDT3	$\bar{\gamma}$	0.004	0.006	0.07	0.01
	$\bar{\Delta}$	0.246	0.248	0.247	0.244
ZDT4	$\bar{\gamma}$	7.251	8.152	17.415	8.962
	$\bar{\Delta}$	0.821	0.635	0.852	0.925
ZDT6	$\bar{\gamma}$	0.003	0.008	0.012	0.021
	$\bar{\Delta}$	0.487	0.493	0.493	0.633

Table 8. Variation of the mean performance metrics vs. the number of particles

Benchmark		n_{mass}		
		50	100	150
ZDT3	$\bar{\gamma}$	**0.005**	**0.005**	0.03
	$\bar{\Delta}$	0.248	**0.247**	0.248
ZDT4	$\bar{\gamma}$	9.241	**7.251**	8.502
	$\bar{\Delta}$	**0.768**	0.821	0.805
ZDT6	$\bar{\gamma}$	**0.003**	**0.003**	0.015
	$\bar{\Delta}$	0.489	**0.487**	0.633

Table 5 and Table 6 compare the performance of NSGSA with that of MOGSA and SMOPSO in the sense of the mean generational distance and spacing metrics, respectively. As can be seen in Table 5, NSGSA has superior convergence in MOP5 and MOPC1 problems than the two other algorithms, while MOGSA has a little better convergence in MOP6. Moreover, NSGSA has better diversity than the two other algorithms in all benchmarks, as compared in Table 6. There are two important reasons that NSGSA has provided better spread in Pareto front. First, an enhanced method to prune the external archive with the limited length is used. Second, some specific members of the archive with higher rank are inserted into

Table 9. Variation of the mean performance metrics vs. the probability of sign mutation

Benchmark		**P_s**		
		0.5	**0.7**	**0.9**
ZDT3	$\bar{\gamma}$	**0.006**	**0.004**	0.005
	$\bar{\Delta}$	0.248	**0.246**	**0.246**
ZDT4	$\bar{\gamma}$	6.015	7.251	10.235
	$\bar{\Delta}$	**0.801**	0.821	0.932
ZDT6	$\bar{\gamma}$	**0.172**	**0.003**	0.004
	$\bar{\Delta}$	0.489	**0.487**	**0.487**

the list of moving particles to search the extreme and the least crowded spaces of the Pareto front.

Sensitivity Analysis

In this section, the sensitivity of NSGSA to its parameters is analyzed based on the average performance, obtained over 10 different runs. In this analysis, three complex benchmarks ZDT3, ZDT4 and ZDT6 are investigated. Table 7 presents the performance of NSGSA for different settings of the inertia weight. As can be seen, the values of 0.9 and 0.5 for w_0 and w_1 are the best for all benchmarks. The large inertia weight motivates exploration and the smaller values motivate exploitation. It can also be observed that the random inertia weight, proposed in the original GSA, does not provide a better performance than the current setting.

Table 8 presents the performance metrics for different values of n_{mass}. As can be seen, the use of 100 particles (current setting) provides the best result. The performance is deteriorated by increasing n_{mass}. However, it is not so sensitive to decreasing n_{mass}.

Table 9 presents the performance metrics for different values of the sign mutation probability (P_s). According to Table 9 the value of 0.7 (current setting) is the best setting for ZDT3 and ZDT6 and the value of 0.5 provides a little better results for ZDT4.

Table 10 presents the performance metrics for different values of the reordering mutation probability (P_r). According to this table, the current setting (0.4) is the best for ZDT3 and ZDT6, while the value of 0.6 seems a better setting for ZDT4.

Table 11 presents the performance metrics for different values of elitism probability ($P_{elitism}$). Again, the current setting (0.5) is the best for all benchmarks. Increasing $P_{elitism}$ motivates the search around the best solutions, found so far and decreases the exploration. Conversely, decreasing $P_{elitism}$ decreases the exploitation. The same results can be concluded from Table 12 for the parameter β.

Table 13 presents the average performance metrics for different values of archive length ($n_{archive}$). The archive length determines the required number of Pareto optimal solutions. Usually, it is not taken as a tuning parameter. The results, presented in Table 13, show that the performance of NSGSA is not so sensitive to this parameter.

The sensitivity analysis, made in this section, shows that NSGSA is not so sensitive to the tuned value of some parameters such as n_{mass} and $n_{archive}$. Moreover, the best value of some other parameters, such as w_0, w_1, $P_{elitism}$ and β, is not so sensitive to the problem at hand.

Table 10. Variation of the mean performance metrics vs. the probability of re-ordering mutation

Benchmark		**P_r**		
		0.2	**0.4**	**0.6**
ZDT3	$\bar{\gamma}$	**0.005**	**0.004**	0.004
	$\bar{\Delta}$	0.248	**0.246**	0.246
ZDT4	$\bar{\gamma}$	10.985	7.251	**6.523**
	$\bar{\Delta}$	**0.923**	0.821	**0.801**
ZDT6	$\bar{\gamma}$	**0.003**	**0.003**	0.102
	$\bar{\Delta}$	0.487	**0.487**	0.487

Table 11. Variation of the mean performance metrics vs. the percent of elitism

Benchmark		**$P_{elitism}$**		
		0.3	**0.5**	**0.7**
ZDT3	$\bar{\gamma}$	0.005	**0.004**	0.02
	$\bar{\Delta}$	0.247	**0.246**	0.249
ZDT4	$\bar{\gamma}$	9.235	**7.251**	11.235
	$\bar{\Delta}$	0.821	**0.821**	0.921
ZDT6	$\bar{\gamma}$	0.007	**0.003**	0.09
	$\bar{\Delta}$	0.487	**0.487**	0.523

CONCLUSION

In this paper, an extension of the single objective GSA to multi-objective optimization problems was presented. The proposed algorithm, called NSGSA, utilizes the non-dominated sorting approach to update the gravitational accelerations. An external archive of the Pareto optimal solutions was also used to provide some elitism. For this purpose, a percent of the archived members are inserted to the list of moving particles. Moreover, to extend the Pareto front and to get a uniform spread of solutions, the extreme and the least crowding members of the external archive are also added to the list of moving particles, respectively.

The new found non-dominated solutions are continually added to the external archive. When the length of the external archive is violated, one member is selected to be removed. For this purpose, a spread indicator, defined as the deviation of the members' crowding distances from the average, was suggested to select a member to be removed. The authors propose the use of this method in other multi-objective algorithms to prune the external archive.

To promote and preserve diversity within the moving particles, two novel mutation operators, called sign and reordering mutations

Table 12. Variation of the mean value of metrics vs. the coefficient of search interval

Benchmark		**β**		
		1.5	**2.5**	**3.5**
ZDT3	$\bar{\gamma}$	0.006	**0.004**	0.005
	$\bar{\Delta}$	0.247	**0.246**	0.248
ZDT4	$\bar{\gamma}$	8.523	**7.251**	10.954
	$\bar{\Delta}$	0.854	**0.821**	0.847
ZDT6	$\bar{\gamma}$	0.06	**0.003**	0.1
	$\bar{\Delta}$	0.488	**0.487**	0.493

were proposed. These operators can also be used in other swarm optimizers such as PSO to improve the diversity within the agents. Moreover, some modifications were applied to the original GSA on the method the gravitational constant is updated.

The performance of NSGSA was tested over a set of benchmark problems, borrowed from the literature, to compare its results with those of the real-coded and binary-coded variants of NSGA-II, MOGSA and SMOPSO. The results show that the new multi-objective variant of GSA, proposed as NSGSA, can obtain comparable and even better performance in the sense of convergence rate and spread of solutions.

Finally, the sensitivity of NSGSA to its parameters was investigated. The results show that the performance is not so sensitive to the value of some parameters and the best value of some parameters is the best for all problems. Therefore, some parameters can be hard coded, if necessary, in order to reduce the number of parameters to tune.

Table 13. Variation of the mean performance metrics vs. archive length

Benchmark		**$n_{archive}$**		
		50	**100**	**150**
ZDT3	$\bar{\gamma}$	0.006	**0.004**	0.006
	$\bar{\Delta}$	**0.244**	0.246	0.245
ZDT4	$\bar{\gamma}$	7.012	**7.251**	7.562
	$\bar{\Delta}$	0.800	0.821	**0.795**
ZDT6	$\bar{\gamma}$	**0.003**	**0.003**	**0.003**
	$\bar{\Delta}$	0.488	**0.487**	0.488

ACKNOWLEDGMENTS

The authors would like to thank Dr. Hossein Nezamabadi-pour for providing the code of single-objective GSA that helped the authors develop the multi-objective variant.

REFERENCES

Cagnina, L., Esquivel, S., & Coello Coello, C. A. (2005). A particle swarm optimizer for multi-objective optimization. *Journal of Computer Science and Technology*, *4*(5), 204–210.

Deb, K., Pratap, A., Agarwal, S., & Meyarivan, T. (2002). A fast and elitist multi-objective genetic algorithm: NSGA-II. *IEEE Transactions on Evolutionary Computation*, *6*(2), 182–197. doi:10.1109/4235.996017

Dorigo, M., Maniezzo, V., & Colorni, A. (1996). The ant system: Optimization by a colony of cooperating agents. *IEEE Transactions on Systems, Man, and Cybernetics – Part B*, *26*(1), 29–41. doi:10.1109/3477.484436

Glover, F. (1989). Tabu search: Part I. *ORSA Journal on Computing*, 1(3), 190-206.

Glover, F. (1990). Tabu search: Part II. *ORSA Journal on Computing*, *2*(1), 4–32. doi:10.1287/ijoc.2.1.4

Hansen, M. P. (1997). Tabu search for multi-objective optimization: MOTS. In *Proceedings of the International Conference of the International Society on Multiple Criteria Decision Making* (pp. 1-16).

Hassanzadeh, H. R., & Rohani, M. (2010). A multi-objective gravitational search algorithm. In *Proceedings of the Second International Conference on Computational Intelligence, Communication Systems and Networks* (pp. 7-12).

Ho, S. L., Shiyou, Y., Guangzheng, N., Lo, E. W. C., & Wong, H. C. (2005). A particle swarm optimization-based method for multi-objective design optimizations. *IEEE Transactions on Magnetics*, *41*(5), 1756–1759. doi:10.1109/TMAG.2005.846033

Holliday, D., Resnik, R., & Walker, J. M. (1993). *Fundamentals of physics*. New York, NY: John Wiley & Sons.

Kennedy, J., & Eberhart, R. C. (1995). Particle swarm optimization. In *Proceedings of the IEEE International Conference on Neural Networks* (Vol. 4, pp. 1942-1948).

Kirkpatrick, S., Gelatto, C. D., & Vecchi, M. P. (1983). Optimization by simulated annealing. *Science*, *220*, 671–680. doi:10.1126/science.220.4598.671

Li, C., & Zhou, J. (2011). Parameter identification of hydraulic turbine governing system using improved gravitational search algorithm. *Journal of Energy Conversion and Management*, *52*, 374–381. doi:10.1016/j.enconman.2010.07.012

Rashedi, E., Nezamabadi-pour, H., & Saryazdi, S. (2009). GSA: A gravitational search algorithm. *Information Sciences*, *179*(13), 2232–2243. doi:10.1016/j.ins.2009.03.004

Sarafrazi, S., Nezamabadi-pour, H., & Saryazdi, S. (2011). Disruption: A new operator in gravitational search algorithm. *Scientia Iranica, Transaction D: Electrical and Computer Engineering*, *18*(3), 539-548.

Srinivas, N., & Deb, K. (1994). Multi-objective optimization using non-dominated sorting in genetic algorithms. *Evolutionary Computation*, *2*(3), 221–248. doi:10.1162/evco.1994.2.3.221

Tang, K. S., Man, K. F., Kwong, S., & He, Q. (1996). Genetic algorithms and their applications. *IEEE Signal Processing Magazine*, *13*(6), 22–37. doi:10.1109/79.543973

Ulungu, E., Teghem, J., Fortemps, P., & Tuyttens, D. (1999). MOSA method: A tool for solving multi-objective combinatorial optimization problems. Journal of Multi criteria. *Decision Analysis*, *8*, 221–236.

Hadi Nobahari was born in Iran in 1976. He received the PhD degree in aerospace engineering from Sharif University of Technology, in 2006. Since 2007 he is an assistant professor in flight dynamics and control. His main research interests are the application of heuristic algorithms in aerospace, intelligent guidance and control systems, and cooperative guidance and navigation of swarms.

Mahdi Nikusokhan was born in Iran in 1980. He received his BS and MS degree in Aerospace Engineering from Sharif University of technology, Iran, in 2002 and 2004 respectively. He is currently a PhD candidate majored in Aerospace Engineering in Sharif University of technology, Iran. His research interests are guidance and control of aerospace vehicles and heuristic optimization.

Patrick Siarry was born in France in 1952. He received the PhD degree from the University Paris 6, in 1986 and the Doctorate of Sciences (Habilitation) from the University Paris 11, in 1994. He was first involved in the development of analog and digital models of nuclear power plants at Electricité de France (E.D.F.). Since 1995 he is a professor in automatics and informatics. His main research interests are computer-aided design of electronic circuits, and the applications of new stochastic global optimization heuristics to various engineering fields. He is also interested in the fitting of process models to experimental data, the learning of fuzzy rule bases, and of neural networks.