Swarm and Evolutionary Computation 14 (2014) 1–14

Contents lists available at ScienceDirect

Swarm and Evolutionary Computation

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

Regular Paper

A gravitational search algorithm for multimodal optimization

Sajjad Yazdani, Hossein Nezamabadi-pour[*], Shima Kamyab

Department of Electrical Engineering, Shahid Bahonar University of Kerman, PO Box 76169-133, Kerman, Iran

A R T I C L E I N F O

Article history:
Received 17 May 2012
Received in revised form
10 August 2013
Accepted 11 August 2013
Available online 20 August 2013

Keywords:
Multimodal optimization
Niching methods
Gravitational search algorithm
Heuristic algorithms

A B S T R A C T

Gravitational search algorithm (GSA) has been recently presented as a new heuristic search algorithm with good results in real-valued and binary encoded optimization problems which is categorized in swarm intelligence optimization techniques. The aim of this article is to show that GSA is able to find multiple solutions in multimodal problems. Therefore, in this study, a new technique, namely Niche GSA (NGSA) is introduced for multimodal optimization. NGSA extends the idea of partitioning the main population (swarm) of masses into smaller sub-swarms and also preserving them by introducing three strategies: a *K*-nearest neighbors (*K-NN*) strategy, an elitism strategy and modification of active gravitational mass formulation. To evaluate the performance of the proposed algorithm several experiments are performed. The results are compared with those of state-of-the-art niching algorithms. The experimental results confirm the efficiency and effectiveness of the NGSA in finding multiple optima on the set of unconstrained and constrained standard benchmark functions.

1. Introduction

Many practical scientific and engineering problems comprise objective functions with multimodal behavior that require optimization methods to find more than one solution. A multimodal problem generally has a number of global optima and several local optima that might be good alternatives to the global ones. On the other hand, the local optima could be excellent alternative solutions in many cases. Therefore, it is desirable in many applications to find the location of all global optima and also other local optima during a search process [1,2]. At least there are two practical reasons for finding multiple optima of an optimization problem [3]. First, by finding multiple optima, the chance of locating the global optimum may be improved. Second, in a design context, identifying a diverse set of high-quality solutions (multiple optima) will give the designer some notion about the nature of the problem, perhaps, may suggest innovative alternative solutions [3].

Traditional heuristic search algorithms like genetic algorithms (GAs) converge towards a single solution (this is the so-called genetic drift phenomenon). GAs perform well in locating a single optimum but fail to provide multiple solutions [2,4], therefore they are not suitable for multimodal optimization. GAs often lose multiple solutions (widely different solutions) due to three effects [5,6]: selection pressure, selection noise, and operator disruption. Unfortunately, other types of heuristic algorithms including standard evolutionary algorithms (e.g. evolutionary programming (EP), evolutionary strategy (ES), differential evolutionary (DE), GA) and standard swarm intelligence (SI) techniques (e.g. ant colony optimization (ACO), particle swarm optimization (PSO)) suffer from the similar disadvantage.

To overcome this problem, many theoretical and empirical studies have been realized to find multiple solutions to multimodal optimization problems using different types of heuristic algorithms especially GAs. Optimization methods that are able to find the location of multiple optima in multimodal problems are known as niching methods [7]. Niching methods partition the population in such a way that each group of the main population focuses on a different possible solution in a single run of a heuristic search. In other words, these methods try to prevent the search from premature convergence and hence aim at preserving diversity in the population and promote the maintenance of stable sub-populations. In the optimization terminology, a niche is referred to as a peak of the fitness landscape, while a species is defined as a subpopulation of similar individuals populating a niche (each subpopulation in the partitioned population).

SI studies the collective behavior of systems composed of many individuals interacting locally with each other and with their environment. Swarms inherently use forms of decentralized control and self-organization to achieve their goals. In SI systems the agents follow very simple rules, although there is no centralized control structure dictating how individual agents should behave. Here, social interactions provide the basis for unguided problem solving. Gravitational Search Algorithm (GSA) is one of the SI-based optimization algorithms which introduced by Rashedi et al. in 2009 based on the metaphor of gravitational interaction

[*] Corresponding author. Tel./fax: +98 341 3235900.
E-mail address: nezam@mail.uk.ac.ir (H. Nezamabadi-pour).

2210-6502/$ - see front matter © 2013 Elsevier B.V. All rights reserved.
http://dx.doi.org/10.1016/j.swevo.2013.08.001

between masses [8]. GSAs are computational models that emulate the law of gravity to solve optimization problems. A GSA comprises a set of mass elements (swarm) and a set of physical inspired operators. According to the Newtonian gravity and motion theories, the population evolves towards increasingly better regions of the search space by simulation of gravity and motion laws. The previous works have revealed the efficiency of the GSA as a global optimizer in solving various nonlinear continuous benchmark functions [8] and its binary version (BGSA) [9] in solving binary encoded problems. Moreover, the results obtained in [10–12] confirm that GSA is a suitable tool for classifier design, parameter identification of hydraulic turbine governing system, and synthesis of thinned scanned concentric ring array antenna, respectively.

Theoretically, GSA belongs to the class of SI-based heuristic algorithms. Rashedi et al. [8] practically gave a comparative study between GSA and a small number of well-known swarm algorithms like PSO. The obtained results reveal that GSA has a merit in the field of optimization. Also, to theoretically highlight the differences between GSA and other heuristic algorithms some distinct features of it has been noted by Rashedi et al. [8].

As mentioned above, GSA artificially simulates the Newton theory that says: every particle in the universe attracts every other particle and the gravity is a force that pulls together all matters. Based on this description, masses in GSA tend to converge towards each other, which in turn means convergence to the global optimum of the problem at hand. Similar to genetic drift in GAs, we refer to this phenomenon as gravity drift. Therefore, a simple GSA is unable to provide multiple solutions in multimodal problems in a single run of the search. Since introducing the standard GSA, there has not been any effort for providing a GSA version for handling multimodal problems. This paper deals with this issue and for this purpose some simple modifications are applied on the standard GSA to support niching in multimodal optimization. In other words, our aim is to show that the standard GSA with a small change is able to effectively handle multimodal problems. For simplicity, here we call the proposed algorithm as Niche GSA (NGSA).

This paper is organized as follows. Section 2 gives us a review on the related works. Section 3 provides a brief introduction to GSA. The proposed Niche GSA for multimodal problems is given in Section 4. The experimental study is illustrated in Section 5, where the performance of the algorithm will be evaluated on a set of benchmark functions. Finally, the paper is concluded in Section 6.

2. Background

Achieving a good method for multimodal optimization by heuristic algorithms will be possible if and only if population diversity is preserved. For the first time, Cavicchio [13] proposed his scheme in 1970, namely as preselection, to preserve the diversity of genetic algorithm. In preselection, each offspring competes with his parents for survival. The better one wins the competition and is transferred to the next population. Substituting the parents by their offspring is the main reason why in this scheme the diversity is preserved. De Jong in his thesis developed the preselection to achieve crowding scheme [14] in which for each offspring, CF (crowding factor) parents are selected at random and the most similar one to the offspring is chosen to be replaced. In crowding, similarity is calculated according to a genotypic measure. It is noted that in ordinary crowding, at each generation only a portion of current population G (generation gap), is selected based on a fitness proportionate strategy to reproduce the next generation.

Both preselection and crowding schemes are unable to find more than two peaks on multimodal problems due to replacement error [15,16]. Mahfoud proposed the deterministic crowding scheme to improve the standard one by reducing the replacement error by the following modifications [17]: (i) individuals are selected for reproduction by random selection strategy, (ii) genotypic similarity measures are replaced by phenotypic one, (iii) each offspring is compared only to its parents and replace the nearest parent if it has a higher fitness value (where in this way offspring and parents of identical niches compete for survival). Unfortunately, deterministic crowding suffers greatly from genetic drift just like its standard version [2].

The authors in [18] described an algorithmic and analytical framework which is applicable to a wide range of crowding algorithms. As an example they analyzed the probabilistic crowding niching algorithm. It is shown that in probabilistic crowding, subpopulations are maintained reliably, and they showed that it is possible to analyze and predict how this maintenance takes place.

Fitness sharing which is the most frequently used scheme for multimodal optimization was first proposed by Goldberg and Richardson [19]. Fitness sharing aims at effective formulation and preservation of stable niches. This scheme has been inspired by nature based on the idea that individuals in a particular niche should share the existing resources. Fitness sharing leads the search in unexplored regions of the search space by artificially decreasing fitness of solutions in crowded areas. According to this idea, the fitness value of a certain solution is degraded proportional to the existing solutions that are located in its neighborhood. Here, the neighborhood is defined in terms of a distance measure and specified by the parameter σ_{share} known as niche radius which is a user defined parameter. To do this, a penalty method is applied to penalize the solutions positioned in populated regions. In other words, for each solution, all other solutions are found in its niche radius and their fitness values are shared using the sharing function. Performance of fitness sharing scheme is highly dependent on value of niche radius which is the same for all peaks. This is the main disadvantage of fitness sharing because each peak needs its own niche radius while similar niche radius for all peaks may result in over- or under-discovering of them.

Mating two individuals from different niches may cause to produce the lethal individuals. The mating restriction scheme was proposed by Deb and Goldberg [20] to promote the effect of fitness sharing scheme. Based on this scheme, two individuals are allowed to mate only if they are within a certain distance of each other (given by the parameter σ_{mating} so-called as mating radius). Mating restriction may avoid the production of lethal individuals and therefore improve the algorithm performance. Sequential niching has been proposed by Beasley et al. in which the niches are discovered sequentially over time [1]. After identifying a niche, the search space is adapted such that to keep away other solutions from the region around the recent discovered solutions. This process is frequently applied in order to focus on unexplored regions and detect undiscovered niches.

In [21], several strategies of sharing have been reviewed and a new recombination schemes has been proposed to improve the efficiency of search algorithm. Finally, the study compares several sharing methods with other niching techniques such as clearing [22]. Among all niching GAs reviewed in this paper, clearing can be considered as the best method [21]. A species conservation genetic algorithm (SCGA) was presented in [3] to evolve parallel subpopulations for multimodal function optimization in which distributed elitism is used where, the population is divided into several species according to their similarity. Each of these species is built around a dominating individual called the species seed. The species seeds found in each generation are conserved by moving them into the next generation. The only difference between the simple GA (SGA) and SCGA is introducing two processes of the selection of seeds and the conservation of species into the main loop of SGA [3]. The proposed method in [23] combines the ideas of SCGA of establishing and conserving a dominating seed for every species with a topological subpopulations separation as in the multimodal genetic algorithm (MGA). On the other hand, besides seed preservation, subpopulations differentiation is performed through the use of the MGA component to distinguish between basins of attraction. Seed dynamics are furthermore controlled and both replication and exploration are concerned [23].

A bi-objective multi-population genetic algorithm (BMPGA) has been recently introduced in [24] aiming to locate all global and local optima on a real-valued differentiable multi-modal landscape. BMPGA uses two separate complementary fitness objectives to enhance the overall population diversity and exploration ability of the algorithm. Using two objectives, individual that are weak in one of fitness objectives but promising in the other are given a chance to survive. They used also a multi-population GA and a clustering scheme to handle multi-modal problems. Clustering is used to help BMPGA to form the desired number of clusters (sub-populations) around potential optima and maintain stable species. In this approach selection is done within the various sub-populations to carry out local evolution of them such that each sub-population is evolved toward its optimum. On the other hand, to prevent extinction of small sub-populations, local elitism is applied [24].

Niching techniques are not restricted to GA. There are many types of niching techniques based on other types of heuristic algorithms like PSO. For example, Brits et al. proposed a version of PSO, called NichePSO, to handle multimodal problems [7]. NichePSO extends the inherent unimodal nature of the standard PSO approach by growing multiple sub-swarms from an initial swarm. In this method, when a possible solution is detected at a particle′s location, a sub-swarm is created. Then the sub-swarm exploits the potential solution. In NichePSO, sub-swarm may merge together, or absorb particles from the main swarm [7]. NichePSO has several parameters that must be specified by user. Speciation-based PSO (SPSO) uses the idea of species [25]. In this algorithm a procedure from previous works has been adopted to determine species and dominant particles in these species. In sequel, each species forms a sub-swarm that can be run as a PSO to exploit the search space around its seed (dominant particle in a species). In this approach, since species are adaptively formed around different optima, multiple global optima can be found [25] through iterations. The quality of SPSO depends on the selection of a niche radius parameter which determines the size of niche or species.

Fitness-Euclidean distance ratio PSO (FER-PSO) is another multi-modal algorithm for finding multiple global optima which was proposed by Li [26]. In this algorithm the concepts of memory-swarm and explorer-swarm have been used to provide a robust multi-modal algorithm. The personal best of particles are used to provide a stable network retaining the best solutions found so far by the swarm as memory-swarm. On the other hand, the current positions of the particles are considered as the explorer-swarm to explore the search space for new solutions. In FER-PSO instead of a single global best, each particle is attracted by a fittest and closest neighborhood solution that is identified via computing its FER value. If the population size is sufficiently large, FER-PSO is able to locate all global optima. The main noticeable advantage of FER-PSO is that it does not require specification of niching parameters [26].

As mentioned before, one of disadvantages of many of niching algorithms is its dependency on niching parameters. This point enforces the user to have a prior knowledge about the problem being optimized, if a high performance results is needed. In order to tackle the need for niching parameter, Li [27] proposed a niching algorithm based on PSO. He showed that the *lbest* PSOs with ring neighborhood topology are able to find the local and global optima in a multi modal problem without needing any niching parameters. He investigates the size of neighborhood (2/3 particles). The experimental results showed that the *lbest* PSO algorithms with an overlapping ring topology are able to locate multiple global optima, given a reasonable large population size, whereas the *lbest* PSO algorithms with a non-overlapping ring topology can be used to locate global as well as local optima, especially for low dimensional problems.

In another work on niching PSO, a distance-based Locally Informed Particle Swarm optimizer (LIPS), has been proposed [28]. LIPS eliminates the need to specify any niching parameter and enhance the fine search ability of PSO. In this algorithm, instead of using the global best particle, LIPS uses several local bests to guide the search of each particle. The experimental results show that LIPS operate as a stable niching algorithm by using the information provided by its neighborhoods. In LIPS the neighborhoods are estimated in terms of Euclidean distance.

Recently a neighborhood mutation strategy has been proposed and integrated with niching differential evolution (DE) algorithms to solve multimodal optimization problems [29]. In this method, the mutation is performed within each Euclidean neighborhood. The experimental results reveal that the neighborhood mutation is able to maintain the multiple optima found during the evolution and evolve toward the respective global/local optimum.

Besides the aforementioned methods for multimodal optimization, there are many widely adopted niching techniques (based on GAs or other types of heuristic algorithms), such as Dynamic niche sharing [5,30], cluster based sharing scheme [16,31–33], dynamic niche clustering that combines clustering and fitness sharing [34,35] adaptive sequential niche [36], coevolutionary sharing [37], crowding clustering approach [2], sharing scheme based on niche identification techniques [15], dynamic fitness sharing [38] and so on. A comprehensive survey on niching algorithms and techniques for multi-modal optimization could be found in [39].

3. Gravitational search algorithm

To make a proper background, the GSA [8] is briefly explained. In GSA, agents are considered as objects and their performance is measured by their masses. All these objects attract each other by a gravity force, and this force causes a movement of all objects globally towards the objects with heavier masses. The heavy masses correspond to good solutions of the problem. Inspired by physics, each mass (agent) has four specifications: its position, its inertial mass, its active gravitational mass, and its passive gravitational mass. Inertial mass, M_i, is a measure of an object′s resistance to changing its state of motion when a force is applied. An object with large inertial mass changes its motion more slowly, and an object with small inertial mass does rapidly. Active gravitational mass, M_a, is a measure of the strength of the gravitational field due to a particular object. Gravitational field of an object with a small active gravitational mass is weaker than an object with a large active gravitational mass. Passive gravitational mass, M_p, is a measure of the strength of an object′s interaction with the gravitational field. Within the same gravitational field, an object with a smaller passive gravitational mass experiences a smaller force than an object with a larger passive gravitational mass.

In GSA the position of the mass corresponds to a solution of the problem, and its gravitational and inertial masses are determined using a fitness function. By lapse of time, masses are attracted by the heaviest mass. This mass will present an optimum solution in the search space. The GSA could be considered as an isolated system of masses. It is like a small artificial world of masses obeying the Newtonian laws of gravitation and motion [8].

Now, consider a system with N agents (masses), the position of the ith agent is defined by:

$$X_i = (x_i^1, \ldots, x_i^d, \ldots, x_i^n) \quad for\ i = 1, 2, \ldots, N \tag{1}$$

where x_i^d presents the position of ith agent in the dth dimension and n is the dimension of search space. At time "t", the force acting on mass "i" from mass "j" is defined as:

$$F_{ij}^d(t) = G(t)\frac{M_{pi}(t) \times M_{aj}(t)}{R_{ij}(t) + \varepsilon}(x_j^d(t) - x_i^d(t)) \tag{2}$$

where M_{aj} is the active gravitational mass related to agent j, M_{pi} is the passive gravitational mass of agent i, $G(t)$ is gravitational constant at time t, ε is a small constant, and $R_{ij}(t)$ is the Euclidian distance between two agents i and j. The total force that acts on agent i in a dimension d is a randomly weighted sum of dth component of the forces exerted from *Kbest* agents:

$$F_i^d(t) = \Sigma_{j \in kbest, j \neq i} rand_j F_{ij}^d(t) \tag{3}$$

where $rand_j$ is a random number in the interval [0,1] and *Kbest* is the set of first K agents with the best fitness value and biggest mass. *Kbest* is a function of time, which is initialized to K_0 at the beginning and decreased with time.

By the law of motion, the acceleration of the agent i at time t, and in direction d, $a_i^d(t)$, is given as follows:

$$a_i^d(t) = \frac{F_i^d(t)}{M_{ii}(t)} \tag{4}$$

where M_{ii} is the inertial mass of ith agent. The next velocity of an agent is considered as a fraction of its current velocity added to its acceleration. Therefore, its velocity and its position could be calculated as follows:

$$v_i^d(t+1) = rand_i \times v_i^d(t) + a_i^d(t) \tag{5}$$

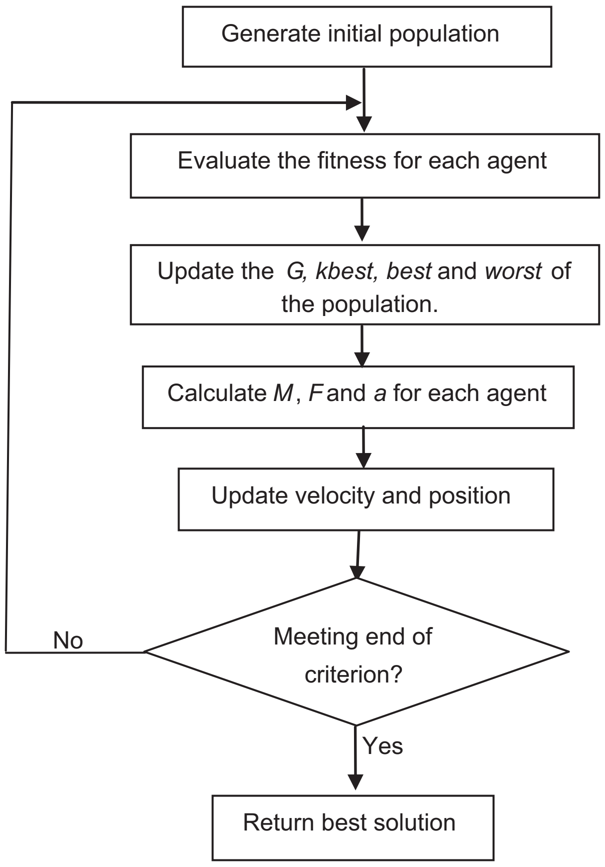
Generate initial population
Evaluate the fitness for each agent
Update the G, kbest, best and worst of the population.
Calculate M, F and a for each agent
Update velocity and position
Meeting end of criterion?
No
Yes
Return best solution

Fig. 1. General principle of GSA.

$$x_i^d(t+1) = x_i^d(t) + v_i^d(t+1) \tag{6}$$

where $rand_i$ is an uniform random variable in the interval [0,1]. The gravitational constant, G, is initialized at the beginning and will be reduced with time to control the search accuracy.

Gravitational and inertia masses are simply calculated by the fitness evaluation. A heavier mass means a more efficient agent. This means that better agents have higher attractions and walk more slowly. Assuming the equality of the gravitational and inertia mass (Eq. (7)), the values of masses are calculated using the map of fitness. The gravitational and inertial masses are updated by the following equations [8]:

$$M_{ai} = M_{pi} = M_{ii} = M_i, \ i = 1, 2, \ldots, N \tag{7}$$

$$q_i(t) = \frac{fit_i(t) - worst(t)}{best(t) - worst(t)} \tag{8}$$

$$M_i(t) = \frac{q_i(t)}{\sum_{j=1}^{N} q_j(t)} \tag{9}$$

where $fit_i(t)$ represents the fitness value of the agent i at t, and, $worst(t)$ and $best(t)$ are defined as follows (for a maximization problem):

$$worst(t) = \min_{j \in \{1,\ldots,N\}} fit_j(t) \tag{10}$$

$$best(t) = \max_{j \in \{1,\ldots,N\}} fit_j(t) \tag{11}$$

The principle of GSA is shown in Fig. 1.

4. The proposed algorithm (NGSA)

Before introducing the proposed niche algorithm, it should be noticed that a multimodal GSA has to be able to tackle three issues including: (i) how to divide the swarm (population) into sub-swarms, (ii) how to preserve these sub-swarms and consequently to avoid the gravitational drift, and (iii) how to connect them to the existing optima within the fitness landscape (i.e. exploit the optimal solutions). In the proposed NGSA, in order to divide the swarm into sub-swarms a K-nearest neighbors (K-*NN*) strategy is proposed, that is, only the K-*NN* of each agent are allowed to apply the gravity force to agent to attract it. Also, a modification on the calculation of the active mass is suggested so that the active mass of each agent does not remain constant for all masses within an iteration of the algorithm. In other words, the active gravitational mass of an agent is changed according to its distance with other masses. On the other hand, to avoid extinction of sub-swarms an elitism strategy is applied while the local implementation of GSA in a sub-swarm provides suitable exploitation ability for fine tuning of a sub-swarm on the existing optimum around it.

In NGSA the main structure and properties of GSA are preserved. However, to give the ability of handling the multimodal problems to GSA, we apply some modification on GSA inspired by fitness sharing and restricted mating in GAs. In fact, the ideas behind GAs are not applicable to the GSA. Therefore, it is necessary to adapt them to be useful in GSA. It is noted that the proposed algorithm is able to form niches and eliminate gravity drift. Before introducing the proposed algorithm let us consider $F(X)$ a typical multimodal n-dimensional real-valued function including m-maxima which should be found. $X = (x^1, x^2, \ldots, x^n)$, $X \in S \subseteq R^n$, S is a bounded set on R^n and $F: S \to R$. According to the above definition, different steps of the proposed NGSA are as follows:

Step 1 (Initialization): Generate randomly the initial swarm of N masses (agents) and set $t = 0$. Each agent represents a candidate solution and is considered as a real-valued vector X_i, $\forall\ i \in \{1, 2, \ldots, N\}$, where X_i s are decision variables to be optimized. The initial values of masses′ components are selected over the predefined decision space S at random using a uniform distribution as follows:

$$x_i^j(0) = x^{j,\min} + U_{ij}(0,1).(x^{j,\max} - x^{j,\min}),\ \forall\ i \in \{1,2,...,N\},\ \forall\ j \in \{1,2,...,n\} \tag{12}$$

where $x^{j,\max}$ and $x^{j,\min}$ denote upper and lower bounds of jth component of decision space, respectively and $U_{ij}(0,1)$ is a random number in the range [0,1] generated by uniform distribution.

Step 2 (Fitness evaluation): Calculate fitness value of each mass $X_i(t)$, $\forall i \in \{1,2,...,N\}$ of the population by computing the objective function at point $X_i(t)$, $fit_i(t)$.

Step3 (Computation of active gravitational mass): Maintaining diversity is a vital factor for supporting multimodal optimization. Therefore, we propose that only the K-nearest neighbors of each agent are able to apply the gravity force to agent to attract it. However, the main difficulty is how one could determine the best number of neighbors. In fact, constructing a robust and effective multimodal GSA is possible only if the masses in the same niche interact with each other by using gravitational force. A simple way is to define a parameter that is referred to as impact radius, σ_{impact}, similar to mating radius in multimodal GAs. However, this makes the algorithm to be problem dependent such that for achieving good results the user shall select the best value for the impact radius. It is noted here that the value σ_{impact} should be selected according to the number of problem′s maxima (niches) and also their distance. Therefore, this is a problem dependent parameter. A large value of σ_{impact} may lose some niches and a small value of σ_{impact} may produce some virtual and non-maxima solutions.

Kennedy has proposed several topologies for describing neighborhood for particles in PSO such as ring, wheels, stars, and random edge [40]. Here, we describe the neighborhood according to a distance in decision space. In other words, those neighbors of each agent that have the smaller distance to that agent based on Euclidian distance are considered as its neighbors. To find K-NN of agent i (KNN_i), distance of all agents to i is calculated and the agents are sorted according to the distance values. Then, the K agents that have the smaller distance to i are selected as neighbors of agent i.

To realize multimodal optimization and prevent gravity drift, we consider the active gravitational mass as a vector containing N elements such that each element corresponds to a mass of the system. In other words, each component determines the effect of the agent to attract another agent of the swarm. In view of this, the active gravitational mass of each agent is presented as a real-valued vector $M_{aj} = [m_{aj1}, m_{aj2}, \ldots, m_{ajN}]$, $\forall\ j \in \{1,2,\ldots,N\}$ where m_{aji} is the gravitational mass of agent j that acts on agent i to attract it which is calculated as:

$$m_{aji}(t) = \begin{cases} \frac{fit_j(t) - worst_i(t)}{best_i(t) - worst_i(t)} & \text{if agent } j \in KNN_i \\ 0 & \text{otherwise} \end{cases} \tag{13}$$

where $fit_j(t)$ represents the fitness value of the agent j at t, and, $worst_i(t)$ and $best_i(t)$ are defined as follows (for a maximization problem):

$$worst_i(t) = \min_{l \in KNN_i} fit_l(t) \tag{14}$$

$$best_i(t) = \max_{l \in KNN_i} fit_l(t) \tag{15}$$

In other words, $worst_i(t)$ and $best_i(t)$ are defined on the neighborhood of agent i at t. In this way, we can control the effect of agent j on agent i according to their distance and quality of agents in the neighborhood of agent i (KNN_i).

Although, we try to design NGSA to be independent of impact radius, selecting the size of neighborhood (the number of neighbors), K, still remains a critical problem which should be discussed. A small value of K leads the algorithm to produce many virtual (non-existing) niches and a large value leads the algorithm to lose some existing niches. To overcome this problem, we adaptively define parameter K as follows:

$$K(t) = Round\left(\left[K_i - (K_i - K_f).\frac{t}{T}\right]N\right) \tag{16}$$

where K_i and K_f are two constants that determine the number of neighbors at the beginning and the end of the search, respectively. When there is no prior knowledge about the number of niches and their distances, in our opinion, the best solution is to initialize K with a small value at the beginning. This will allow the niches to form and then we grow the neighborhood radius by increasing the number of neighbors by time. This will find the most possible niches of the problem at hand by merging the near virtual niches to a real niche ($0 < K_i < K_f < 1$).

Step 4 (Calculation of acceleration, velocity and position): At time t, the force acting on mass i from mass j is defined as:

$$F_{ij}^d(t) = G(t)\frac{M_{pi}(t) \times m_{aji}(t)}{R_{ij}(t) + \varepsilon}(x_j^d(t) - x_i^d(t)) \tag{17}$$

The total force that acts on agent i in the dimension d is computed as follows:

$$F_i^d(t) = \sum_{j \neq i} rand_j F_{ij}^d(t) \tag{18}$$

However, the acceleration of agent i and its next velocity in the dimension d are computed according to Eqs. (4), and (5), respectively. It should be noted that we consider $M_{pi} = M_{ii}$, $\forall\ i \in \{1,2,...,N\}$.

After calculating the velocity, the next position of each agent, $x_i'^d(t)$, could be computed according to (19). After that, the fitness value of solutions $X'_i(t)$, $\forall\ i \in \{1,2,...,N\}$ are calculated and then the next position is updated according to Eq. (20)) (for a maximization problem) which is similar to elitism strategy in the EAs, for each agent the new position will be accepted if it provides a solution with higher quality.

$$x_i'^d(t) = x_i^d(t) + v_i^d(t+1) \tag{19}$$

$$X_i(t+1) = \begin{cases} X'_i(t) & \text{if } fit(X'_i) \geq fit(X_i) \\ X_i(t) & \text{otherwise} \end{cases} \tag{20}$$

If we do not consider the elitism strategy, when K is large, it may cause some masses which are in the scope of a small niche to be attracted by high quality masses of large niches. This causes the algorithm to miss some niches.

Step 5: $t = t+1$, Steps 3–5 are repeated until the stopping criterion is satisfied.

The main difference between canonical GSA and the proposed NGSA are summerized as follows. The $K-NN$ strategy (Eqs. (13)–(16)) is used for computing active gravitational mass. This strategy prevents the problem of gravitational drift. The elitism strategy (Eq. (20)) is used for updating the position of agents; this is effective in the preserving the sub-swarms. By these changes we have added two parameters K_i and K_f to the algorithm instead of parameter K in the canonical GSA.

5. Experimental results

To evaluate the performance of the proposed NGSA, it is applied on several widely used multi-modal benchmark functions of different characteristics, such as deceptiveness, multiple evenly and unevenly spaced global optima, and multiple global optima in the presence of multiple local optima. These functions have been recently used in [27]. The experiments are carried out by following three aims. First, we try to understand the behavior of the proposed NGSA by some experiments on a few number of test functions. Second, a comparative study is done to assess the effectiveness of the NGSA in locating multiple global optima and finally to show the effectiveness of NGSA in finding both global and local optima in test functions. Thus, the performance of NGSA is compared to those of state-of-the-art algorithms including r2-PSO [27], r3-PSO [27], r2PSO-lhc [27], r3PSO-lhc [27], SPSO [25], FER-PSO [26], NichePSO [7], deterministic Crowding [17], and sequential niche [1], and the results are reported in the following.

5.1. Multi-modal benchmark functions

NGSA is applied on a number of multi-modal functions (Table 1) where, the aim is to discover peaks by considering two cases; (i) finding all global maxima, and (ii) finding both local and global maxima. Functions F_1–F_5 were firstly introduced by Goldberg and Richardson to evaluate their fitness sharing approach [19] and thereafter they were used by other researches such as Beasley et al. to test their sequential niching algorithm [1], Mahfoud to examine Niching GAs [41], and Brits et al. for evaluation of NichePSO [7]. Also, other functions have been used by many researches to test and compare their methods for solving multi-modal problems. The locations of all optima of these functions are known. This fact makes them ideal for evaluating the performance of heuristic search algorithms for discovering niches in multi-modal domains. Due to this characteristic, one can compare the final distribution of solutions obtained by the algorithm with the ideal population distribution. The functions F_1–F_4 are one-dimensional and are defined over the interval [0,1]. Each of F_1 and F_2 has five maxima at 0.1, 0.3, 0.5, 0.7, and 0.9. For F_3 and F_4, their maxima are located at 0.08, 0.25, 0.45, 0.68, and 0.93. As depicted in Fig. 2, F_5 is a two-dimensional function containing four maxima which are located at (3.0, 2.0), (−3.87, −3.28), (3.58, −1.85), and (−2.81, 3.13) [7]. F_6–F_8 are one dimensional functions with unbalance distribution of maxima location. These functions are deceptive and difficult because the existing local optima can misguide the population to move away from the true global optimum [27]. For example in F_6 the global optima is located at $x=20$ that has only 1/4 of the initial solutions in its scope. Besides this, the locations of both maximum points of F_6 are in the outskirt of the scope of the solutions. Function F_9 is a modification of Six-Hump Camel Back function that was introduced by Tamson. This function is used by Quing [2] and Li [27] as a benchmark. This function has 2 global optima and two local optima.

Function F_{10} has 25 evenly spaced maxima of unequal heights with one being equal to the global maximum. If the goal of optimization is to find all global and local optima then this function can be used to test the ability of the algorithm to locate all 25 maxima. The functions F_{11} and F_{12} can be solved in a different dimension. Fig.2 illustrates these functions in a 2-dimensional space. In this scenario, F_{11} has 760 local maxima and 18 global maxima [27]. For the n-dimensional inverted Shubert function, there are $n \times 3^n$ global maxima unevenly distributed. These global peaks are divided into 3^n groups, with each group having n global peaks that are close to each other. Function F_{12} in a 2-dimensional case has 36 optima with similar values but unevenly spaced. For more details regarding these functions, the readers are referred to [27]. In our experiments all 12 functions are

Table 1
Test functions used in the experiments.

Name	**Test function**	**Range**	Number of Global peaks	Number of all peaks
Equal maxima	$F_1(x) = \sin^6(5\pi x)$	$0 \le x \le 1$	5	5
Decreasing maxima	$F_2(x) = e^{-2\log(2)\times\left(\frac{x-0.1}{0.8}\right)^2} \sin^6(5\pi x)$	$0 \le x \le 1$	1	5
Unevenmaxima	$F_3(x) = \sin^6(5\pi(x^{\frac{3}{4}} - 0.05))$	$0 \le x \le 1$	5	5
Uneven decreasing maxima	$F_4(x) = e^{-2\log(2)\times\left(\frac{x-0.08}{0.854}\right)^2} \sin^6(5\pi(x^{\frac{3}{4}} - 0.05))$	$0 \le x \le 1$	1	5
Himmelblau's function	$F_5(x_1, x_2) = 200 - (x_1^2 + x_2 - 11)^2 - (x_1 + x_2^2 - 7)^2$	$-6 < x_1, x_2 < 6$	4	4
Two-peak trap	$F_6(x) = \begin{cases} \frac{160}{15}(15-x) & \text{for } 0 \le x \le 15 \\ \frac{200}{5}(x-15) & \text{for } 15 \le x \le 20 \end{cases}$	$0 \le x \le 20$	1	2
Central two-peak trap	$F_7(x) = \begin{cases} \frac{160}{10}x & \text{for } 0 \le x \le 10 \\ \frac{160}{5}(15-x) & \text{for } 10 \le x \le 15 \\ \frac{200}{5}(x-15) & \text{for } 15 \le x \le 20 \end{cases}$	$0 \le x \le 20$	1	2
Five-uneven-peak-trap	$F_8(x) = \begin{cases} 80(2.5-x) & \text{for } 0.0 \le x \le 2.5 \\ 64(x-2.5) & \text{for } 2.5 \le x \le 5.0 \\ 64(7.5-x) & \text{for } 5.0 \le x \le 7.5 \\ 28(x-7.5) & \text{for } 7.5 \le x \le 12.5 \\ 28(17.5-x) & \text{for } 12.5 \le x \le 17.5 \\ 32(x-17.5) & \text{for } 17.5 \le x \le 22.5 \\ 32(27.5-x) & \text{for } 22.5 \le x \le 27.5 \\ 80(x-27.5) & \text{for } 27.5 \le x \le 30 \end{cases}$	$0 \le x \le 30$	2	5
Six-Hump Camel Back	$F_9(x_1, x_2) = -4\left[\left(4 - 2.1x_1^2 + \frac{x_1^4}{3}\right)x_1^2 + x_1x_2 + (-4 + 4x_2^2)x_2^2\right]$	$-1.9 \le x_1 \le 1.9$ $-1.1 \le x_2 \le 1.1$	2	4
Shekel's foxholes	$F_{10}(x_1, x_2) = 500 - \frac{1}{0.002 + \sum_{i=0}^{24} \frac{1}{1 + i + (x_1 - a(i))^6 + (x_2 - b(i))^6}}$ where $a(i) = 16((i \bmod 5) - 2)$, and $b(i) = 16(\lfloor(i/5)\rfloor - 2)$	$-65.536 \le x_1, x_2 \le 65.535$	1	25
Inverted Shubert function	$F_{11}(x) = -\prod_{i=1}^{n} \sum_{j=1}^{5} j \times \cos[(j+1)x_i + j]$	$-10 \le x_i \le 10$	$n \times 3^n$	a
Inverted Vincent function	$F_{12}(x) = \frac{1}{n}\sum_{i=1}^{n} \sin(10 \times \log(x_i))$	$0.25 \le x_i \le 10$	6^n	6^n

*For inverted Shubert with $n=2$, there are 760 local peaks [27].

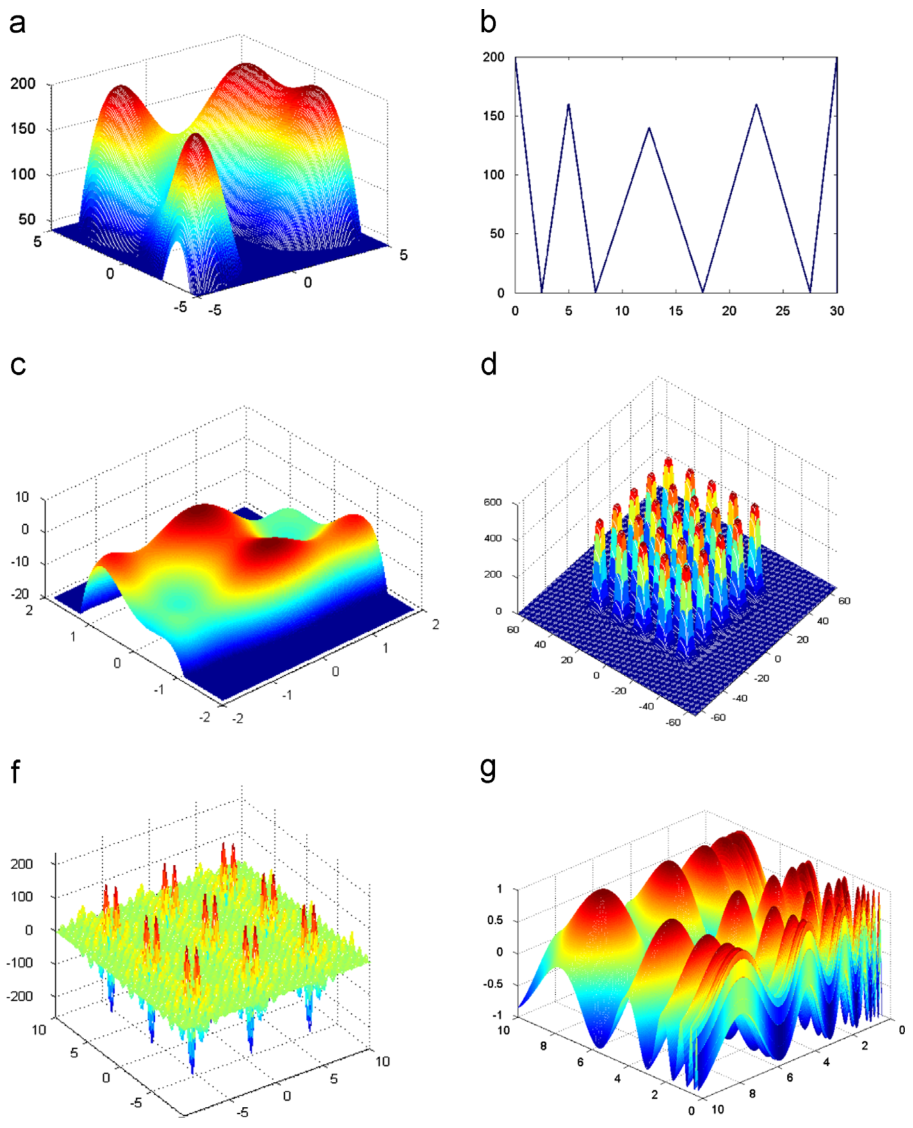

Fig. 2. Some benchmark functions (a) F_5, (b) F_8, (c) F_9, (d) F_{10}, (e) F_{11} and (f) F_{12}.

used to evaluate the ability of the proposed NGSA in finding the global solutions and functions F_1 to F_{10} are used to find all optima including global and local ones.

5.2. *Performance criteria*

The results obtained by NGSA are compared with those available in the literature. The comparison is made in terms of the successful discovering rate of all actual niches of the problem during the evolution. In other words, a run is considered as a success if the algorithm could correctly find all possible peaks of the problem. A peak is considered detected if an agent approaches %99 of its peak value. The results are averaged over many independent runs (e.g. 30 or 50) and the average discovering rate (ADR) are reported for NGSA. Also, we count the number of fitness evaluations in finding all peaks and report its average value over independent runs as a performance criterion.

In addition, we define another performance measure to evaluate the exploitation ability of the proposed NGSA algorithm. Consider the function to be optimized containing m niches in an n-dimensional space, i.e. $S_i=(s_i^1, s_i^2, \dots, s_i^n)$, $i=1,2,\dots,m$ and the algorithm could discover the final solutions $\widehat{S}_i=(\widehat{s}_i^1, \widehat{s}_i^2, \dots, \widehat{s}_i^n)$, $i=1,2,\dots,m$. In this case, we define an error criterion for a successful run as:

$$\zeta=\frac{1}{m}\sum_{i=1}^{m}\left(\sum_{j=1}^{n}(s_i^j-\widehat{s}_i^j)^2\right)^{1/2} \tag{21}$$

ζ is calculated in the last iteration and are averaged over successful runs of many independent runs (e.g. 30 or 50) of the algorithm. Here, the smaller values of ζ means the better performance of the algorithm.

5.3. *Preliminary results*

We applied the NGSA on the maximization multi-modal functions shown in Table 1 with the aim of finding all maxima including local and global maxima. In all cases, the maximum iteration T is considered to be 120. Based on our experiments, K_i and K_f in Eq. (16) are set to 0.08 and 0.16, respectively. We evaluated the effect of population size in this stage. In this regard, we ran the experiments using several values for N such as 20, 35, 50, and 75. In NGSA (Like GSA [8]), $G(t)$ is set using Eq. (22), where G_0 is set to $0.1\times D_f$ and α is set to 8, where D_f is the domain of the function to be optimized.

$$G(t) = G_0 e^{-\alpha(t/T)} \tag{22}$$

To have a better clarity, the results obtained by NGSA with different population numbers for each function are averaged over independent runs. The discovering rate (DR) of each run in the last iteration is recorded and averaged (ADR) over 30 independent runs. The obtained ADR for functions F_1 to F_5 are given in Table 2.

As shown by Table 1, the proposed algorithm is able to detect all actual maxima when $N=75$. Also, the results obtained by $N=50$ are satisfactory. Experimental results show that for all functions, better results are obtained for NGSA with a bigger population size.

As mentioned before, all masses in an artificial system tend to converge towards each other. According to our idea, NGSA try to localize the convergence and divide the population into sub-swarms such that each one converges to a niche. The main problem in case of small population size is when around a specific niche there are not enough masses in the randomized initialization. In this case, the algorithm is not able to provide a sub-swarm around the corresponding niche. Therefore, the niche is missed. For example, NGSA with $N=20$ is not able to discover all niches of function F_1 in %20 of runs.

To get a robust performance in finding all optima using NGSA, we have two solutions. One is to select sufficient number of masses (population size) where it leads the algorithm to discover all optima by producing sufficient sub-swarms, each one around a niche. Another way is to try uniformly distributing the initial population over the search landscape. In order to do that, we partition the search space into equal interval and then randomly distribute the masses through the partitions. In the next subsection we examine this issue.

Table 2
%ADR obtained by the NGSA with different population size, and random initialization to find all global and local maxima.

Function	$N=75$	$N=50$	$N=35$	$N=20$
F_1	100	100	93	80
F_2	100	100	96	73
F_3	100	100	86	73
F_4	100	96	90	66
F_5	100	100	96	76

Table 3
%ADR obtained by the NGSA with different population size and partition based initialization to find all global and local maxima.

Function	$N=75$	$N=50$	$N=35$	$N=20$
F_1	100	100	100	100
F_2	100	100	100	100
F_3	100	100	100	100
F_4	100	100	100	100
F_5	100	100	100	100

5.4. NGSA with partition based initialization

In this manner, at the beginning the search space is deterministically partitioned into equal size intervals (areas). Thus, we take the same approach as proposed in [42] to partition the search landscape. Next, we randomly put the masses throughout the intervals. In this case, we set all parameters like the previous stage and repeat the experiments. The results obtained are illustrated in Table 3.

These results show that the algorithm could successfully discover all niches (including local and global) of the benchmark functions F_1 to F_5, independently of the population size. We also calculated the results in the case of $N=20$, $T=120$, for functions F_1 to F_5 based on error ζ. For this, the algorithm is run 30 times and ζ is computed in the last iteration and recorded for each run, and then averaged over independent runs. The results are summarized and reported in Table 4. In this table the best results obtained during 30 runs of the algorithm are also reported.

Figs. 3 and 4 show how the NGSA works and depict its ability in finding and preserving all peaks of the fitness landscape. Fig. 3 shows the fitness landscape for the one-dimensional function F_4 in four iterations: $t=1$, 10, 20, and 30 where N, T, K_i and K_f are set to 20, 40, 0.08, and 0.16, respectively. As this figure shows just in 30 iterations the population find all peaks. In this figure, the sings "+", "*" and "○"stand for position X_i, position X'_i and the peaks which are detected by the algorithm. Fig. 3b shows that the algorithm could find a close solution to all 5 maxima (both local and global) at iteration 10 and it completely converges to their final values at iteration 30.

As Fig. 4 shows, we have 4 sub-swarms for function F_5 at iteration $t=1$ in which each sub-swarm is depicted by a specific sign. With the lapse of time, each sub-swarm will converge to the nearest niche or peak of the fitness landscape. However, it can be seen from Fig. 4b that some masses may move between different sub-swarms. In other words, there are interactions between species in the swarm and it is possible that a mass escapes from the scope of one niche and can be attracted by another niche. However, as Fig. 4d shows, the proposed NGSA finds all optima and converge to them at $t=100$.

5.5. Sensitivity to parameter K

It is obvious that the performance of the NGSA highly depends on how the active gravitational mass is calculated. Based on Eq. (13), computing the active gravitational mass is affected by the number of neighbors, K, which is defined by Eq. (16). To see why K must be varied with time (Eq. (16)), we performed some experiments with K as a constant according to

$$K = Round(K_0 N) \tag{22}$$

where K_0 takes different values in the interval [0.1, 0.3].

The setup of these experiments such as population size, maximum iteration, etc. is the same as Sections 4 and 5.The results are computed in terms of the number of discovered niches

Table 4
The average error ζ obtained by the NGSA with $N=20$, $T=120$ and partition based initialization.

Function	Average ζ	Best result				
		S_1	S_2	S_3	S_4	S_5
F_1	$1.62e-5 \pm 3.47e-5$	0.1000	0.3000	0.5000	0.7000	0.9000
F_2	$0.001 \pm 1.66e-4$	0.1000	0.2999	0.4998	0.7000	0.8979
F_3	$0.0012 \pm 5.04e-4$	0.0797	0.2466	0.4506	0.6814	0.9339
F_4	$2.37e-4 \pm 3.38e-4$	0.0797	0.2463	0.4495	0.6790	0.9302
F_5	0.0570 ± 0.0262	(3.0001, 2.0008)	(−3.7800, −3.2841)	(3.5829, −1.8514)	(−2.8130, 3.1377)	–

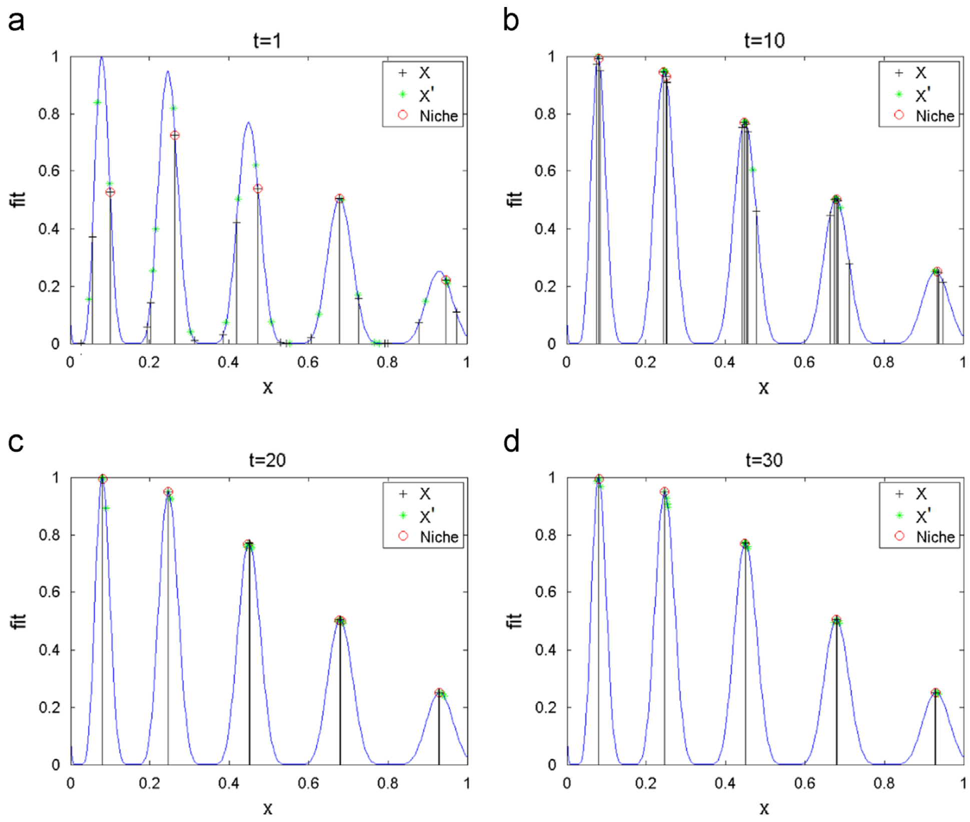

Fig. 3. Niching behavior induced by proposed NGSA on F_4 at iterations (a) $t=1$ (b) $t=10$ (c) $t=20$ (d) $t=30$.

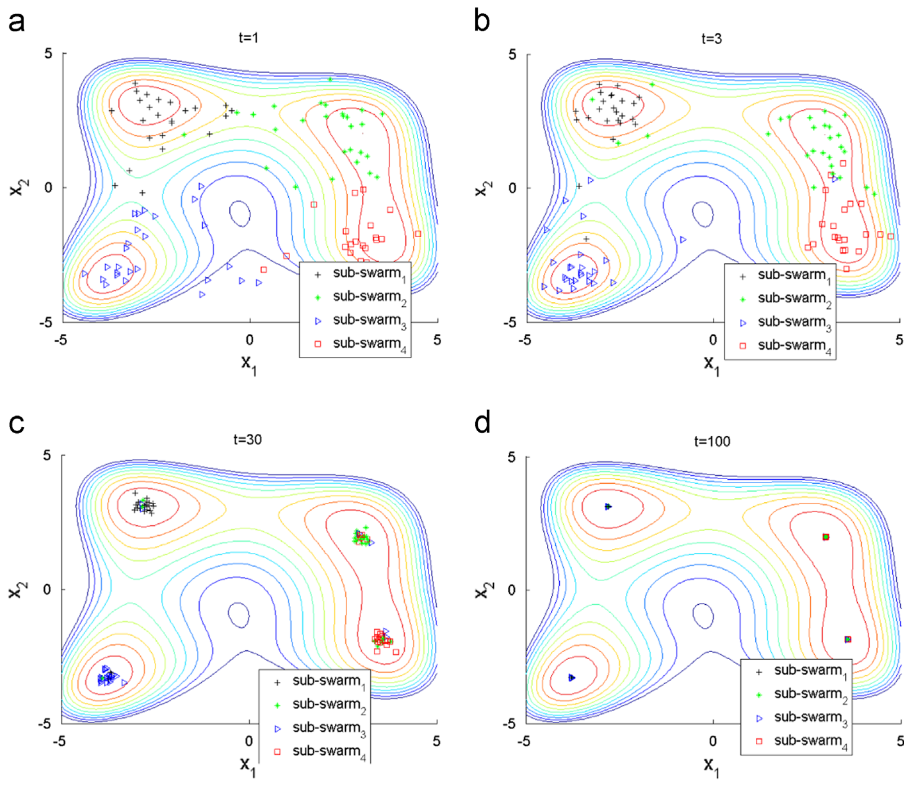

Fig. 4. Niching behavior induced by the proposed NGSA on F_5 at iterations (a) $t=1$ (b) $t=3$ (c) $t=30$ (d) $t=100$.

by the algorithm, which are averaged over 30 independent runs for functions F_1 to F_5. The results are presented by Fig. 5. As this figure shows, the small values of K_0 increase the number of discovered niches and generate virtual niches. On the other hands, large values of K_0 make the algorithm to lose some actual niches (peaks of the fitness landscape). Also, the number of maxima, m,

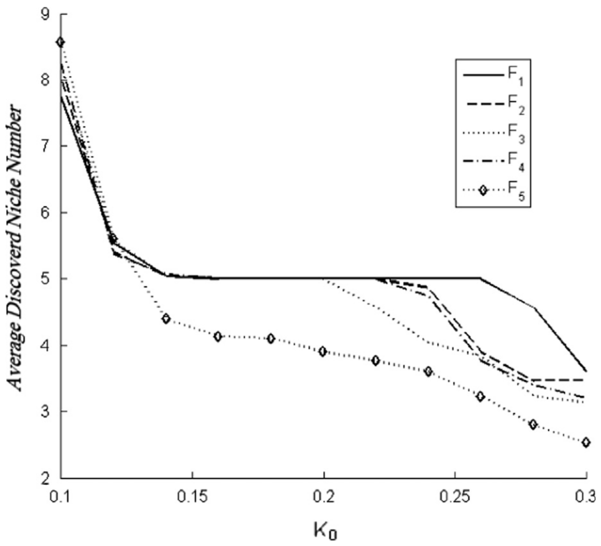

Fig. 5. The effect of number of neighbors on the performance of the algorithm in terms of average number of discovered niches for functions F_1 to F_5. The experiments setup is $N=20$, $T=120$ and partition based initialization. K_0 is constant during each run. The results are averaged over 30 independent runs.

plays an important role on selecting K_0. This fact is inferred by Fig. 5 for functions F_1–F_4 which have 5 maxima in contrast to function F_5 containing 4 maxima.

We can conclude that K is a problem dependent parameter which plays an important role on the performance of the algorithm and it is dependent on the number of niches of the problem. Referring to Eq. (16), we start the search with a small value of K (K_i), which is linearly increased with time to K_f at the end of the search process. This can tackle the lack of knowledge about the actual number of niches and allows us to follow the search with a prediction of the number of niches. We found that to achieve good results for all functions, the suitable values for K_i and K_f are $0.8/m$ and $0.95/m$, respectively. If we know the niche radius, r, based on our experiments, the suitable values for K_i and K_f are $1.6r/D_f$ and $1.9r/D_f$, respectively.

To investigate why K should be varied with time (Eq. (16)), we performed an experiment on the F_{10}. For better clarification, the fitness landscape and the position of masses for F_{10} are presented for several iterations in Fig. 6. As Fig. 6a illustrates at $t=1$ when the size of K is 3, the algorithm finds many virtual peaks. With the lapse of time and with the increase of K the number of virtual niches are reduced and many of them are merged together so that they converge towards the actual peaks of the fitness landscape. Fig. 6d shows that the NGSA find all peaks of the function by creating and preserving sub-swarms around each niche.

5.6. Comparative study

The comparison is categorized in two groups. The first one shows a comparative study to assess the effectiveness of the NGSA in locating multiple global optima where the performance of NGSA is compared to those of state-of-the-art algorithms including r2-PSO [27], r3-PSO [27], r2PSO-lhc [27], r3PSO-lhc [27], SPSO [25], and FER-PSO [26] that have been already reported by Li in [27]. The second one shows the effectiveness of NGSA in finding both global and local optima in test functions. NGSA is applied on the first 10 functions and the results obtained for F_1 to F_5 are compared with those obtained by NichePSO [7], deterministic Crowding [17], and sequential niche [1] that have been reported by Brits et al. in [7].

Case 1. Finding global maxima.

The obtained results for F_1 to F_{12} with regard to ADR are summarized in Table 5. The results for NGSA are averaged over 50 independent runs. In this case, all settings are the same as reported in [27]. The only difference is for functions $F6$ to $F9$ that we used the population size of 100 while in [27] the population size of 50 was used. To make the comparisons meaningful, in NGSA we decreased the maximum number of iterations for these functions to have the same evaluations number as stopping criterion. The population sizes for NGSA are reported in Table 5.

Table 5 states ADR as the percentage of the performance consistency of the tested niching algorithms. Performance consistency reflects the algorithm′s ability to consistently locate all solutions for each of the optimization problems [7]. Table 5 shows that all comparative algorithms are not able to sufficiently preserve solution sets. Overall, the worst results belongs to SPSO, it generally performed worse than the average whereas the best results are for the proposed NGSA. It is worth noticing that the results obtained by r3PSO and r3PSO-lhc are very close to those of NGSA. The NGSA find and locate all global solutions for all cases except for three cases including F_8, F_{11}(3D) and F_{12}(1D) where in these cases the ADR values are 94%, 94% and 92%, respectively. It is worthwhile mentioning that for F_{12}(1D), although NGSA could not provide ADR=100%, it performs better than other comparative algorithms except r2PSO which it results in ADR=94%. In total, the average of ADR over all tested functions is 98.46% which is better than all other comparative algorithms. The closest algorithm to NGSA, is r3PSO which provide the average value of 96% under the same conditions.

Table 6 presents the average numbers of fitness function evaluations required for niching algorithms to converge to all global maxima. These results are connected to the results presented in Table 5. Based on the results given in Table 6 we can compare the computational cost of each of the comparative algorithms in terms of the number of evaluations of fitness functions required to converge to all solution sets. For the values reported in Table 6, in the format $\mu \pm \sigma$, (μ=mean and σ=standard deviation), the following conclusion can be summarized:

- NGSA required fewer evaluations on functions F_1, F_3, F_6, F_7, F_8, F_{11}(2D), F_{11}(3D) and F_{12}(1D) whereas SPSO provided fewer evaluations on functions F_2, F_4, and F_5, r2PSO-lhc required fewer evaluations on function F_9, and r3PSO provide fewer evaluations on function F_{10}.
- Overall, the proposed NGSA required the least number of fitness function evaluations to converge to all solution sets.
- SPSO required more fitness function evaluations than the other considered algorithms.
- Among r2PSO, r3PSO, r2PSO-lhc and r3PSO-lhc, it is r3PSO which needs the least number of fitness evaluations to converge and it is the closest algorithm to the proposed NGSA.
- The NGSA required 2067 (average on all functions) fitness evaluations over all cases whereas the closest algorithms to it, r3PSO, required 13068 (average on all functions) evaluations number which is 6 times greater than that of NGSA.

Tables 5 and 6 illustrate that with the trade-off between accuracy and computational cost, the NGSA is quite better than comparative algorithms. Considering only the ADR measure, the results obtained by the proposed NGSA in all cases are comparable with r3PSO and r3PSO-lhc algorithms and in some cases are better than others.

Case 2. Finding global and local maxima.

Brits et al. have compared their algorithm NichePSO [7] with deterministic Crowding [17], sequential niche [1]. The results

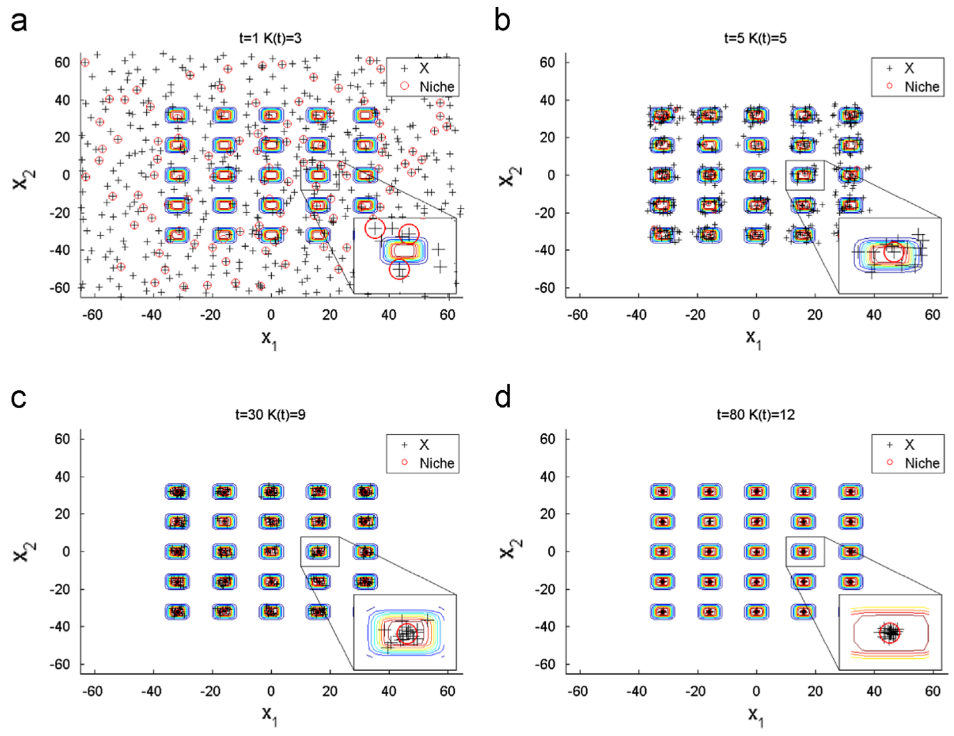

Fig. 6. The niching behavior induced by proposed NGSA on F_{10} at iterations (a) $t=1$ (b) $t=3$ (c) $t=30$ (d) $t=80$. In this figure the sings "+" and "O"stand for position X_i and the peaks found out by the algorithm. The experiments setup is $N=500$, $T=100$ and partition based initialization.

Table 5
Comparison of NGSA, r2PSO, r3PSO, r2PSO-lhc, r3PSO-lhc, FER-PSO, and SPSO in case of finding all global maxima in terms of %ADR for Functions F_1 to F_{12}. The results for r2PSO, r3PSO, r2PSO-lhc, r3PSO-lhc, FER-PSO, and SPSO have been already reported in [27]. The results are averaged over 50 independent runs of algorithm.

Function	N	k_i	k_f	NGSA (%)	r2PSO (%)	r3PSO (%)	r2PSO-lhc (%)	r3PSO-lhc (%)	FER-PSO (%)	SPSO (%)
$\boldsymbol{F_1}$	50	0.08	0.16	**100**	**100**	**100**	**100**	**100**	**100**	**100**
$\boldsymbol{F_2}$	50	0.2	0.4	**100**	98	**100**	**100**	**100**	**100**	**100**
$\boldsymbol{F_3}$	50	0.08	0.16	**100**	98	98	**100**	**100**	**100**	**100**
$\boldsymbol{F_4}$	50	0.2	0.4	**100**	**100**	**100**	**100**	**100**	**100**	**100**
$\boldsymbol{F_5}$	50	0.08	0.16	**100**	92	74	**100**	98	98	**100**
$\boldsymbol{F_6}$	100	0.2	0.4	**100**	98	**100**	94	78	88	24
$\boldsymbol{F_7}$	100	0.3	0.5	**100**	**100**	96	98	88	**100**	22
$\boldsymbol{F_8}$	100	0.2	0.3	94	**100**	96	96	96	98	40
$\boldsymbol{F_9}$	100	0.15	0.4	**100**	**100**	**100**	**100**	**100**	**100**	**100**
$\boldsymbol{F_{10}}$	500	0.01	0.04	**100**	**100**	**100**	72	78	**100**	50
$\boldsymbol{F_{11}}$**(2D)**	250	0.03	0.09	**100**	90	98	98	**100**	56	49
$\boldsymbol{F_{11}}$**(3D)**	500	0.02	0.07	94	4	**100**	4	92	0	0
$\boldsymbol{F_{12}}$**(1D)**	100	0.02	0.03	92	**94**	86	92	90	88	84
Average				**98.46**	90.31	96.00	88.77	93.85	86.77	66.85

Table 6
Average number of fitness function evaluations required to converge for each niching algorithm for results presented in Table 5. The results for r2PSO, r3PSO, r2PSO-lhc, r3PSO-lhc, FER-PSO, and SPSO have been already reported in [27]. The results are averaged over 50 independent runs of algorithm.

Function	NGSA	r2PSO	r3PSO	r2PSO-lhc	r3PSO-lhc	FER-PSO	SPSO
F_1	**263** ± **89**	376 ± 30	443 ± 51	396 ± 51	447 ± 52	384 ± 29	355 ± 30
F_2	300 ± 107	2120 ± 1999	141 ± 11	143 ± 14	144 ± 13	170 ± 12	**127** ± **9**
F_3	**334** ± **81**	2430 ± 1994	2440 ± 1994	456 ± 33	623 ± 273	371 ± 31	343 ± 23
F_4	316 ± 76	175 ± 17	160 ± 20	178 ± 18	162 ± 16	189 ± 20	**144** ± **13**
F_5	1632 ± 330	7870 ± 2891	21400 ± 5467	1490 ± 138	7380 ± 3347	5070 ± 1945	**1250** ± **45**
F_6	**477** ± **409**	3460 ± ± 197	2620 ± 874	7390 ± 3348	23200 ± 5834	14400 ± 4535	77200 ± 5859
F_7	**243** ± **108**	2960 ± 1520	5340 ± 2764	4340 ± 2229	13100 ± 4588	2110 ± 227	78300 ± 5856
F_8	**694** ± **853**	978 ± 186	4650 ± 2784	4710 ± 2783	6730 ± 3088	2660 ± 1992	63300 ± 6773
F_9	1032 ± 892	619 ± 24	684 ± 30	**618** ± **30**	650 ± ± 25	965 ± 53	653 ± 32
F_{10}	4164 ± 1768	4360 ± 559	**3510** ± **453**	29700 ± 6277	24800 ± 5738	3470 ± 336	42800 ± 6968
F_{11}(2D)	**5369** ± **1930**	55900 ± 2676	39100 ± 1648	37800 ± 1480	32400 ± 581	94900 ± 1261	61600 ± 4463
F_{11}(3D)	**9913** ± **2187**	199000 ± 830	74000 ± 2343	198000 ± 1789	81300 ± 5849	200000 ± 0	200000 ± 0
F_{12}(1D)	**2134** ± **430**	8310 ± 3371	15400 ± 4906	9600 ± 3824	14700 ± 4344	13000 ± 4601	17000 ± 5192
Average	**2067.00**	22196.77	13068.31	22678.54	15818.15	25976.08	41774.77

reported on functions F_1 to F_5 in the case of finding all maxima (local and global). We used the setup of Sections 4 and 5 for NGSA the same as that of Brits et al. [7] (for example the population size is 20). Tables 7 and 8 summarize the performance of the niching algorithms. The results obtained by the proposed NGSA in all cases are comparable with NichePSO, Sequential Niche and Deterministic Crowding, and in some cases are better than them.

The results given in Table 7 show that all considered algorithms are not able to sufficiently preserve solution sets. Overall, the worst results belong to Sequential Niche, whereas the best ones are for the proposed NGSA with regards to ADR. It is worth noticing that the results obtained by Niche PSO are close to those of NGSA. NGSA could find and locate all solutions including local and global peaks for all functions reported in Table 7. On the other hand, Table 8 reveals that NGSA required fewer evaluations on the functions F_1, to F_5, with respect to other comparative algorithms. Tables 7 and 8 illustrate that the NGSA is better than the comparative algorithms in terms of ADR and computational cost.

Also the results obtained for functions F_6 to F_{10} are given in Table 9. For these functions we used the same setup of Case 1 for NGSA. As it is shown by Table 9, the NGSA locate all global and local peaks in F_6, F_7 and F_{10}. Also it is able to locate all peaks of F_8 and F_9 with a rate of 94% and 98%. Overall, considering functions F_1 to F_{10} the NGSA locate and preserve all peaks of functions with 99.2%. Furthermore, the results obtained in Table 9 show that the average ζ is in an acceptable range for a multimodal algorithm.

5.7. *Performance comparison using constrained multimodal benchmark functions*

In order to evaluate the performance of NGSA in finding global optima in complex constrained fitness landscapes we used 3 constrained multimodal benchmark functions F_{13}, F_{14} and F_{15} as described in Table 10. The functions F_{13} and F_{14} proposed in [43], called cRastrigin and cGriewank, are constrained multimodal problems by applying equality constraints into Rastrigin and Griewank basic multimodal benchmark functions, respectively. For F_{13} we used two-dimensional version of cRastrigin in experiments in which there are 2 feasible maxima in the domain approximately at $(-4.5, 4.5)$, $(4.5, -4.5)$. In the case of F_{13} the initial population is constructed with the procedure described in [43]. For F_{13} the parameters of NGSA are set as $N=300$, $T=200$, $k_i=0.003$, and $k_f=0.006$.

The function F_{14} has rugged fitness landscape with 4 global maxima approximately at $(-512, 512)$, $(512, -512)$, $(512, 512)$ and $(-512, -512)$. Each member of the initial population is constructed according to the procedure described in [43]. For this function, the parameter values of NGSA are chosen as $N=200$, $T=100$, $k_i=0.003$, and $k_f=0.006$.

The function F_{15} is a constrained multimodal function which proposed by Deb and Saha in [44] called CMMP (n, G, L) with n variable, I constraints, G global and L local minimum points. The values $I=1$, $n=2$, $G=1$ and $L=3$ are used in our experiments to evaluate the performance of NGSA in finding a global optimum at $(-3, 3)$ in presence of 3 local optima in a constrained multimodal fitness landscape. The coefficients a_1, a_2, b_1, b_2 are set to -0.1417, 0.4, 0.2 and -0.6 respectively and $c=2$ is considered in experiments. The parameters of NGSA for F_{15} are set to $N=200$, $T=100$, $k_i=0.015$, $k_f=0.045$.

For handling constraint in the search process we used the repair approach in all methods in which once an unfeasible solution is generated, the initialization procedure of each function is used to replace the generated solution to a feasible one.

The performance of NGSA is compared with local best PSO variants [27] and deterministic crowding algorithms as powerful PSO based and GA based multimodal optimization methods. The experimental results (averaged over 50 independent runs) are shown in Table 11. The results confirm the effectiveness and efficiency of NGSA in finding accurate global optima in constrained multimodal fitness landscapes.

Table 7
Comparison of NGSA, NichePSO, Sequential Niche and Deterministic Crowding algorithms in terms of %ADR for Functions F_1 to F_5. The results for NichePSO, Sequential Niche, and Deterministic Crowding have been reported in [7]. The results are averaged over 30 independent runs of algorithm.

Function	k_i	k_f	NGSA (%)	Niche PSO (%)	Sequential Niche (%)	Deterministic Crowding (%)	Average (%)
F_1	0.08	0.16	**100**	**100**	**100**	**100**	100
F_2	0.08	0.16	**100**	93	83	93	92.25
F_3	0.08	0.16	**100**	**100**	**100**	90	97.50
F_4	0.08	0.16	**100**	93	93	90	94.00
F_5	0.08	0.16	**100**	**100**	86	90	94.00
Average			**100**	97.20	92.40	92.60	–

Table 8
Average number of fitness function evaluations required to converge for each niching algorithm for functions F_1 to F_5. The results for NichePSO, Sequential Niche, and Deterministic Crowding have been already reported in [7]. The results are averaged over 30 independent runs of algorithm.

Function	NGSA	Niche PSO	Sequential Niche	Deterministic Crowding	Average
F_1	**1786 ± 204**	2372 ± 109	4102 ± 577	14647 ± 4612	5726
F_2	**1892 ± 561**	2934 ± 475	3505 ± 463	13052 ± 2507	5345
F_3	**1752 ± 273**	2404 ± 195	4141 ± 554	13930 ± 3284	5556
F_4	**1806 ± 307**	2820 ± 517	3464 ± 287	13929 ± 2996	5504
F_5	**2033 ± 201**	2151 ± 200	3423 ± 402	14296 ± 3408	5475
Average	**1853**	2536	3727	13970	–

Table 9
Results of NGSA in terms of %ADR and the number of fitness evaluations required to converge for each niching algorithm for Functions F_6 to F_{10}. The results are averaged over 50 independent runs of algorithm.

Function	N	k_i	k_f	ADR (%)	Average ζ	Number of fintness evaluation
F_6	100	0.10	0.35	100	4.27e−4 ± 4.18e−4	542 ± 386
F_7	100	0.20	0.45	100	7.44e−6 ± 6.01e−6	321 ± 118
F_8	100	0.10	0.25	94	2.19e−3 ± 1.37e−2	966 ± 765
F_9	100	0.05	0.35	98	1.50e−3 ± 3.33e−2	1122 ± 353
F_{10}	500	0.01	0.03	100	9.66e−4 ± 1.18e−3	6186 ± 2133
Average				98.4	1.02e−3	1827.40

6. Conclusions

In recent years, various heuristic optimization algorithms have been developed. GSA is a new heuristic search algorithm that is constructed based on the laws of gravity and motion. The GSA algorithm uses the theory of Newtonian physics and its searcher agents are the collection of masses. In this paper, a version of GSA called NGSA was introduced for multi-modal optimization. In order to evaluate our algorithm, we tested it on a set of various standard benchmark functions. NGSA has been compared with a number of well-known alternative approaches. In the proposed NGSA, in order to divide the swarm into sub-swarms, a *K*-nearest neighbors (*K-NN*) strategy is proposed where only the *K-NNs* of each agent are allowed to apply the gravity force to agent to attract it. A modification on the calculation of the active gravitational mass is suggested so that the active gravitational mass of each agent does not remain constant for all masses within an iteration

Table 10
Constrained Test functions used in the experiments.

Name	Test function	Range	Number of Global peaks	Number of all peaks
cRastrigin function	$F_{13}(x)=10n+\sum_{i=1}^{n}[x_i^2-10\cos(2\pi x_i)]$ $s.t.\ h(x)=\sum_{i=1}^{n}x_i=0$	$-5.12 \le x_i \le 5.12$	2	2
cGriewank function	$F_{14}(x)=\frac{1}{4000}\sum_{i=1}^{n}x_i^2-\prod_{i=1}^{n}\cos(\frac{x_i}{\sqrt{i}})+1$ $s.t.\ h(x)=\frac{1}{512(n-1)}\sum_{i=2}^{n}x_i^2-\lvert x_1\rvert=0$	$-512 \le x_i \le 512$	4	4
Deb′s constrained function	$F_{15}(x)=(x_1+a_1)^2+(x_2+a_2)^2+\cdots+(x_n+a_n)^2$ $s.t.\ n^2[(x_1+b_1)^2+\cdots+(x_n+b_n)^2]\ge n^2,$ $-(n+1)\le x_i \le n+1$	$-3 \le x_i \le 3$	1	4

Table 11
The results of performance comparison on constrained test functions.

Test function	methods	ADR(%)	Number of fitness evaluations	Average ζ
F_{13}	**r2PSO**	100	510 ± 438.1082	$3.1586e-04 \pm 0.0080$
	r3PSO	100	546 ± 396.0056	$6.3425e-04 \pm 0.0105$
	r2PSO-lhc	100	474 ± 383.7569	$5.5256e-04 \pm 0.0067$
	r3PSO-lhc	100	450 ± 292.2468	$1.2619e-03 \pm 0.0083$
	Deterministic crowding	100	1098 ± 662.8602	$4.3717e-04 \pm 0.0103$
	NGSA	100	$\mathbf{78 \pm 146.0919}$	$\mathbf{3.1255e-04 \pm 0.0107}$
F_{14}	**r2PSO**	100	2396 ± 0.1469	$2.3313e-02 \pm 2.2522$
	r3PSO	100	2092 ± 0.5050	$\mathbf{7.1531e-03 \pm 0.9835}$
	r2PSO-lhc	100	2476 ± 0.5123	$8.9047e-03 \pm 1.8314$
	r3PSO-lhc	100	2232 ± 0.5539	$1.3116e-02 \pm 2.1065$
	Deterministic crowding	100	21552 ± 1.0056	$1.6902e-02 \pm 0.4956$
	NGSA	100	$\mathbf{1944 \pm 1.2944e+03}$	$1.9672e-02 \pm 0.3790$
F_{15}	**r2PSO**	100	$\mathbf{788 \pm 0.0849}$	0.3966 ± 0.2352
	r3PSO	100	792 ± 0.0849	0.381 ± 0.1997
	r2PSO-lhc	100	812 ± 0.0396	$4.9832e-03 \pm 0.2081$
	r3PSO-lhc	100	796 ± 0.0480	0.3237 ± 0.1873
	Deterministic crowding	100	6672 ± 0.0283	$6.2571e-04 \pm 0.0552$
	NGSA	100	3116 ± 1.0135	$\mathbf{5.0262e-04 \pm 0.0203}$

of the algorithm. Moreover, to avoid extinction of sub-swarm we applied elitism while the local implementation of GSA in a sub-swarm provides suitable exploitation ability for fine tuning of a sub-swarm on the existing optimum around it. The results obtained by NGSA have been compared with state-of-the-art algorithms in this field. The results showed that the NGSA is a proper niche algorithm in finding both global optima and local optima. The results suggest that the proposed approach which is inspired by the law of gravity has a merit in the field of multi-modal optimization. However, much work has to be done to establish how well NGSA performs against other complex cases problems such as what given in [45] as the future research. Here, we used the concepts of *K-NN* and elitism to avoid the effect of niche radius. For wide acceptability of the algorithm and also applicability in engineering domains for black box optimization problems, an empirical rule should be suggested for parameter *K*; this makes the algorithm robust and independent of the specific problem. This also can be considered as scope for future study.

Acknowledgement

The authors would like to thanks the anonymous reviewrs for their helpful comments. Also, they would like to extend their appreciation to Dr. S.R. Seydnejad and Dr. M.M. Farangi for proof reading the manuscript and providing valuable comments.

References

[1] D. Beasley, D. Bull, R.R. Martin, A sequential niche technique for multimodal function optimization, Evolutionary Computation 1 (2) (1993) 101–125.
[2] L. Qing, W. Gang, Y. Zaiyue, W. Qiuping, Crowding clustering genetic algorithm for multimodal function optimization, Applied Soft Computing 8 (2008) 88–95.
[3] J.-P. Li, M.E. Balazs, G.T. Parks, P.J. Clarkson, A Species conserving genetic algorithm for multimodal function optimization, Evolutionary Computation 10 (3) (2002) 207–234.
[4] E. Zitzler, L. Thiele, Multi-objective evolutionary algorithms: a comparative case study and the strength Pareto approach, IEEE Transaction on Evolutionary Computation 3 (4) (1999) 257–271.
[5] B.L. Miller, M.J. Shaw, Genetic algorithms with dynamic niche sharing for multimodal function optimization, in: Proceedings of the IEEE International Conference on Evolutionary Computation, Piscataway, NJ, 1996, pp. 786–91.
[6] E..Zitzler, Evolutionary Algorithms for Multi-objective Optimization: Methods and Applications, Ph.D. Thesis, Swiss Federal Institute of Technology Zurich, 1999.
[7] R. Brits, A.P. Engelbrecht, F. Van den Bergh, Locating multiple optima using particle swarm optimization, Applied Mathematics and Computation 189 (2007) 1859–1883.
[8] E. Rashedi, H. Nezamabadi-pour, S. Saryazdi, GSA: a gravitational search algorithm, Information Sciences 179 (13) (2009) 2232–2248.
[9] E. Rashedi, H. Nezamabadi-pour, S. Saryazdi, BGSA: binary gravitational search algorithm, Natural Computing 9 (3) (2010) 727–745.
[10] A. Bahrololoum, H. Nezamabadi-pour, H. Bahrololoum, M. Saeed, A prototype classifier based on gravitational search algorithm, Applied Soft Computing 12 (2) (2012) 819–825.
[11] C. Li, J. Zhou, Parameters identification of hydraulic turbine governing system using improved gravitational search algorithm, Energy Conversion and Management 52 (1) (2011) 374–381.
[12] A. Chatterjee, G.K. Mahanti, Comparative performance of gravitational search algorithm and modified particle swarm optimization algorithm for synthesis

of thinned scanned concentric ring array antenna, Progress In Electromagnetics Research B 25 (2010) 331–348.
[13] D.J.Cavicchio, Adaptive Search Using Simulated Evolution, Ph.D. Thesis, University of Michigan, 1970.
[14] K.A. De Jong, An Analysis of the Behavior of a Class of Genetic Adaptive Systems, Ph.D. Thesis, University of Michigan, 1975.
[15] C.-Y. Lin, W.-H. Wu, Niche identification techniques in multimodal genetic search with sharing scheme, Advances in Engineering Software 33 (2002) 779–791.
[16] A. El Imrani, A. Bouroumi, H. Zine El Abidine, M. Limouri, A. Essaid, A fuzzy clustering-based niching approach to multimodal function optimization, Journal of Cognitive Systems Research 1 (2000) 119–133.
[17] S.W. Mahfoud, Crossover interactions among niches, in: Proceedings of the First IEEE Conference on Evolutionary Computation, Orlando, FL, USA, 1 (Jun. 1994), pp. 188–193.
[18] O.J. Mengshoel, D.E. Goldberg, The crowding approach to niching in genetic algorithms, Evolutionary Computation 16 (3) (2008) 315–354.
[19] D.E. Goldberg, J. Richardson, Genetic algorithms with sharing for multimodal function optimization, in: Proceedings of the Second International Conference on in Genetic Algorithms and Their Applications, John J. Grefenstette (Ed.), L. Erlbaum Associates Inc., Hillsdale, NJ, 1987, pp. 41–49.
[20] K. Deb, D.E. Goldberg, An investigation of niche and species formation in genetic function optimization, in: Proceedings of the Third International Conference on Genetic Algorithms, J. David Schaffer (Ed.), Morgan Kaufmann George Mason University, Fairfax, Virginia, USA, Jun. 1989, pp. 42–50.
[21] B. Sareni, L. Krahenbuhl, Fitness sharing and niching methods revisited, IEEE Transactions on Evolutionary Computation 2 (3) (1998) 97–106.
[22] A. Petrowski, A clearing procedure as a niching method for genetic algorithms, in: Proceedings of Third IEEE International Conference on Evolutionary Computation, J. David Schaffer (Ed.), Nagoya, Japan, May 1996, pp. 798–803.
[23] C. Stoean, M. Preuss, R. Stoean, D. Dumitrescu, Multimodal optimization by means of a topological species conservation algorithm, IEEE Transactions on Evolutionary Computation 14 (6) (2010) 842–864.
[24] J. Yao, N. Kharma, P. Grogono, Bi-objective multi population genetic algorithm for multimodal function optimization, IEEE Transactions on Evolutionary Computation 14 (1) (2010) 80–102.
[25] D. Parrott, X. Li, Locating and tracking multiple dynamic optima by a particle swarm model using speciation, IEEE Transaction on Evolutionary Computation 10 (4) (2006) 440–458.
[26] X. Li, Multimodal function optimization based on fitness-Euclidean distance ratio, in: Proceedings of the Ninth Annual Conference on Genetic and Evolutionary Computation (GECCO '07), ACM, New York, NY, USA, 2007, pp. 78–85. DOI: 10.1145/1276958.1276970.
[27] X. Li, Niching without niching parameters: particle swarm optimization using a ring topology, IEEE Transactions on Evolutionary Computation 14 (1) (2010) 150–169.
[28] B.Y..Qu, P.N. Suganthan S. Das, A distance-based locally informed particle swarm model for multi-modal optimization, IEEE Transaction on Evolutionary Computation, (2013) 10.1109/TEVC.2012.2203138.
[29] B.Y. Qu, P.N. Suganthan, J.J. Liang, Differential evolution with neighborhood mutation for multimodal optimization, IEEE Transaction on Evolutionary Computation 15 (5) (2012) 601–614.
[30] X. Liu, H. Liu, H. Duan, Particle swarm optimization based on dynamic niche technology with applications to conceptual design, Advances in Engineering Software 38 (2007) 668–676.
[31] X. Yin, N. Germany, A fast genetic algorithm with sharing scheme using cluster analysis methods in multimodal function optimization, in: Proceedings of IEEE International Conference on Artificial Neural Networks and Genetic Algorithms, Innsbruck, Austria, 1993, pp. 450–457.
[32] C.-Y. Lin, Y.-J. Yang, Cluster identification techniques in genetic algorithms for multimodal optimization, Computer-Aided Civil and Infrastructure Engineering 13 (1) (1998) 53–62.
[33] J. Alami, A. El Imrani, A. Bouroumi, Multi-population cultural algorithm using fuzzy clustering, Applied Soft Computing 7 (2007) 506–519.
[34] J. Gan, K. Warwick, A genetic algorithm with dynamic niche clustering for multimodal function optimization, in: Proceedings of Fourth IEEE International Conference on Artificial Neural Network and Genetic Algorithms, 1999, pp. 248–255.
[35] J. Gan, K. Warwick, Dynamic niche clustering: a fuzzy variable radius niching technique for multimodal optimization in GAs, in: Proceedings of IEEE Congress on Evolutionary Computation, Seoul, Korea, vol. 1, May 2001, pp. 215–222.
[36] J Zhang, D.-S. Huang, T.-M. Lok, M.R. Lyu, A novel adaptive sequential niche technique for multimodal function optimization, Neurocomputing 69 (2006) 2396–2401.
[37] D.E. Goldberg, L. Wang, Adaptative niching via coevolutionary sharing, in: D. Quagliarella, et al., (Eds.), Genetic Algorithms in Engineering and Computer Science, Wiley, Chichester, 1997, pp. 21–38.
[38] A.D. Cioppa, C. De Stefano, A. Marcelli, Where are the niches? dynamic fitness sharing, IEEE Transaction on Evolutionary Computation 11 (4) (2007) 453–465.
[39] S. Das, S. Maity, B.Y. Qu, N. Suganthan, Real-parameter evolutionary ultimodal optimization—a survey of the state-of-the-art, Swarm and Evolutionary Computation 1 (2011) 71–88.
[40] J. Kennedy, Small worlds and mega-minds: effects of neighborhood topology on particle swarm performance, in: IEEE Proceedings of the Congress of Evolutionary Computation (CEC 99), Washington, DC, USA, vol. 3, Jul. 1999, pp. 1931–1938.
[41] S.W. Mahfoud, Niching Methods for Genetic Algorithms, Ph.D. Dissertation, Univ. Illinois, Urbana-Champaign, 1995.
[42] R.A. Formato, Central force optimization: A new metaheuristic with applications, in: Applied Electromagnetics Progress in Electromagnetics Research 77, 2007, pp. 425–491.
[43] S..Kimura, K..Matsumura, Constrained Multimodal Function Optimization using a Simple Evolutionary Algorithm, IEEE Congress on Evolutionary Computation, New Orleans, LA, USA, (5–8, June 2011, pp. 447–454.
[44] K. Deb, A. Saha, Finding multiple solutions for multimodal optimization problems using a multi-objective evolutionary approach, in: Proceedings of the 12th Annual Conference on Genetic and Evolutionary Computation, 2010, pp. 447–454.
[45] B.Y..Qu P.N..Suganthan, Novel Multimodal Problems and Differential Evolution with Ensemble of Restricted Tournament Selection, IEEE Congress on Evolutionary Computation, Barcelona, Spain, July 2010 3480-3486.