Cognitive Computation (2019) 11:719–734
https://doi.org/10.1007/s12559-019-09665-9

Rank-Based Gravitational Search Algorithm: a Novel Nature-Inspired Optimization Algorithm for Wireless Sensor Networks Clustering

Sepehr Ebrahimi Mood [1] · Mohammad Masoud Javidi [1]

Received: 12 December 2018 / Accepted: 3 July 2019 / Published online: 18 July 2019

Abstract
Recently, wireless sensor networks (WSNs) have had many real-world applications; they have thus become one of the most interesting areas of research. The network lifetime is a major challenge researched on this topic with clustering protocols being the most popular method used to deal with this problem. Determination of the cluster heads is the main issue in this method. Cognitively inspired swarm intelligence algorithms have attracted wide attention in the researh area of clustering since it can give machines the ability to self-learn and achieve better performance. This paper presents a novel nature-inspired optimization algorithm based on the gravitational search algorithm (GSA) and uses this algorithm to determine the best cluster heads. First, the authors propose a rank-based definition for mass calculation in GSA. They also introduce a fuzzy logic controller (FLC) to compute the parameter of this method automatically. Accordingly, this algorithm is user independent. Then, the proposed algorithm is used in an energy efficient clustering protocol for WSNs. The proposed search algorithm is evaluated in terms of some standard test functions. The results suggest that this method has a better performance than other state-of-the-art optimization algorithms. In addition, simulation results indicate that the proposed clustering method outperforms other popular clustering method for WSNs. The proposed method is a novel way to control the exploration and exploitation abilities of the algorithm with simplicity in implementation; therefore, it has a good performance in some real-world applications such as energy efficient clustering in WSNs.

Keywords Wireless sensor network (WSN) · Energy efficient protocol · Clustering method · Gravitational search algorithm (GSA) · Rank-based selection · Fuzzy logic controller (FLC)

Introduction

Biologically inspired algorithms (BIAs) have attracted considerable research interest from science and engineering communities around the world. These algorithms are motivated by challenges in applications where classical optimization algorithms are not effective. Evolutionary algorithms and swarm intelligence are two main branches of BIAs [1].

In recent years, investigation of the behavior of swarm animals and natural phenomena has introduced many metaheuristic algorithms. These algorithms which are inspired by social cognitive science are used to solve optimization problems in real-life applications [2]. However, these algorithms do not have their origin primarily in mathematics; they are inspired by biological phenomena. In BIAs and swarm intelligence algorithms, agents share their information with other agents in the population in order to improve the performance of the whole swarm. Although each agent in these approaches has low-level cognition and is not individually smart enough to solve sophisticated problems, there are some valuable properties such as thinking, learning, recognition [3], reasoning, and high-cognition in the swarm intelligence algorithms which can be used to solve many complicated and industrial problems [4, 5]. There have been numerous advances in swarm intelligence based on cognitively inspired algorithms [6, 7]. Cognitively inspired swarm intelligence algorithms (SIAs) have attracted wide attention in many scientific and engineering fields [8–11]. In SIAs, particles usually either make decisions according to their own experience or according to the elite's suggestions. Moreover, the stochastic mechanism is introduced to all SIAs to promote their optimization capability [12].

Gravitational search algorithm (GSA) is one of the new nature-inspired optimization algorithms which has been proposed by Rashedi et al. [13, 14]. This method is inspired by the Newtonian law of motion, gravity, and mass interactions.

✉ Mohammad Masoud Javidi
javidi@uk.ac.ir

[1] Department of Computer Science, Faculty of Mathematics and Computer, Shahid Bahonar University of Kerman, Kerman, Iran

In this method, agents or masses can search for feasible areas in order to find better solutions. They are placed in a feasible area where their weight is determined by their position in this area and their fitness function value. Thus, the objects which are placed in a better position, in the feasible area, are heavier and attract lighter objects with worse fitness values. GSA is used in applications such as clustering, robotics, filter modeling, and artificial intelligence [15].

Up to date, numerous efforts have been made to enhance the performance of the GSA. Clustered-GSA (CGSA), inspired by calculating the central mass of a structure in nature, is used to reduce the computation of GSA. Further, this method reduces the number of fitness function calculations [16]. Also, some researchers have added a new operator to improve the performance of GSA. Doraghinejad et al. introduced an operator called "Black-hole" for GSA which is inspired by characteristics of black-holes as astronomical phenomenon [17]. Disruption is another operator inspired by astrophysics. This operator can balance and control the exploitation and exploration properties of GSA [18]. Some of the different variants of GSA which have been developed to enhance and improve the original version of the algorithm have been reviewed in here [15]. In this paper, we use the rank-based fitness assignment method to overcome the scaling problems of the proportional fitness assignment. Rank-based methods tend to prevent premature convergence by tempting selection pressure for large fitness differentials that occur in early generations. Conversely, by amplifying small fitness differences in later generations, selection pressure is enhanced compared to other fitness assignment strategies.

BIAs and meta-heuristic methods have the problem of parameter adaptation. Many researchers use a fuzzy logic controller (FLC) to solve this problem and obtain better results than original algorithms [19]. In this regard, Ref. [20] used the fuzzy logic to combine the result of real genetic algorithm (RGA) and particle swarm optimization (PSO) which are biologically inspired optimization algorithms, in an optimal way to achieve the best performance. Ref. [21] investigated a new fuzzy PSO method with-cross mutated operation. This method improved the performance of the algorithm by controlling parameters of the method and the new operator. This approach was used to compute a decision-based filter for a cDNA microarray image restoration. In this paper, a fuzzy logic controller was used to control the selection pressure of the algorithm according to the computational process of GSA. In other words, the exploitation and exploration abilities of the algorithm can be controlled automatically.

In our previous works [18, 22], we attempted to improve the performance of the GSA by introducing new versions of GSA. The authors in [18] proposed a modified version of GSA, with a new operator and mutation, which used the fuzzy logic controller to control the parameters of this operator. In GSA, the value of masses was calculated with respect to the fitness of agents. To enhance the performance of GSA, definition and calculation of masses are necessary. Accordingly, [22] proposed new methods for defining and calculating the value of masses. They proposed mass scaling and Boltzmann selection functions to calculate the value of the masses. Thereby, exploration and exploitation properties of the algorithm could be controlled and its performance improved. In these methods, the value of masses for each agent was heavily dependent on the value of the objective function and fitness function.

In past years, researchers have focused on wireless sensor networks (WSNs) in both theoretical and industrial fields, as they are effective means in monitoring and tracking applications. For example, WSNs are used for health monitoring, military applications, agriculture, forecasting, and decision systems [6, 23]. WSNs contain many cheap independent nodes known as sensor nodes. Each sensor node has units including sensing, processing, communication, and energy units. These sensors are located in the target area randomly or manually. They can sense data such as pressure, humidity, sounds, and chemical concentration from the target area. Sensor nodes collect these data and communicate with other nodes to ultimately relay information to the base station.

The power unit of sensor nodes is limited. Further, in most applications, sensor nodes are not accessible once they are located in the target space. Therefore, their power unit and the battery cannot be changed, and the sensor node will die once its battery runs out. Thus, energy consumption is an issue for WSN investigations in order to enhance the lifetime of the entire network.

Clustering methods have been used in many real-world applications and problems. Hence, it is the fundamental topic in artificial intelligence. In [24], the authors proposed a new clustering protocol called the spectral embedded clustering framework and used linearity regularization to deal with out-of-sample data. Ref. [10] used a social recognition-based multi-objective GSA, for clustering of remote sensing imagery. One of the most common techniques for energy efficient consumption in WSNs is clustering. A clustering method gathers the sensor nodes into clusters, where each cluster has its own cluster head. The nodes sense the data from the target area and transmit them to their corresponding cluster head. Thus, the raw information of all sensors in each cluster is collected into the cluster head. Finally, the cluster head compresses and transmits this information to the base station. Clustering methods can improve the energy usage in the WSNs by reducing the number of nodes with long distance communications and preventing them from sending irrelevant data to the base station.

Low energy adaptive clustering hierarchy (LEACH) is one of the most popular clustering methods for WSNs, first proposed by Heinzelman in [25]. LEACH cannot provide an ideal and desired number of cluster heads in WSNs architecture. A centralized energy efficient clustering method, LEACH-C, was introduced by Muruganathan et al. [26]. In this protocol, the base station forms the clusters at the beginning of each round using a centralized algorithm. The results reveal that LEACH-C protocol can outperform the LEACH algorithm and has a better performance in terms of the network lifetime and total data transmission. Ref. [27] studied some popular clustering methods in WSNs and examined their performance in terms of power consumption.

BIAs and swarm intelligence algorithms have been used in some clustering protocols for WSNs [28]. For example, in [29], the authors defined a new cost function which could minimize intra-cluster node distances and improve the battery usage of the network simultaneously. Then, PSO algorithm was used to find a good solution for this cost function. In [30], a new version of GSA, quadrivalent quantum-inspired gravitational search algorithm (QQIGSA), inspired by quantum mechanics, was introduced, where this optimization algorithm was used in WSNs in order to improve the energy consumption in the network. In Ref. [6], a new, cognitively inspired artificial bee colony clustering algorithm was introduced. This algorithm which has a clustering evaluation model, was used in clustering for energy management in cognitive WSNs, special models of WSNs. The objective function defined in this paper tries to improve the energy consumption in WSNs by determining compact clusters with high remaining energy cluster heads. In the proposed method, we use the modified version of GSA to find a good solution for this optimization problem.

In this paper, a new method is used for mass calculation in GSA where a rank-based approach considers the rank of individuals instead of their fitness values is employed to determine the value of masses. The exploration and exploitation of the algorithm can be controlled by the parameter of the algorithm. In the proposed method, a fuzzy controller is defined to determine the value of this parameter; therefore, the parameter is determined automatically and independent of the user. Further, this controller can balance the exploitation and exploration abilities of the algorithm to achieve the best performance for solving optimization problems. The performance of the proposed method is compared with state-of-the-art heuristic algorithms. The experimental results and statistical analysis reveal that the proposed method outperforms other approaches in many test functions. Reviews of the literature show that the proposed method is the first application of a rank-based method in mass calculation in GSA. Also, the use of a controller to determine the value of this parameter and to control the abilities of the algorithm has not been previously done. This novel version of GSA was then used in WSN protocols to optimize the battery usage and maximize the network lifetime. The simulation results suggest that the proposed clustering method has a better performance than other popular clustering algorithms. This is particularly important as this application of GSA in energy efficient clustering problems is the first of its kind.

The paper is organized as follows: the “Basic Concepts” section introduces the basic concepts required to present the proposed method; “Rank-Based GSA” section displays the improved version of GSA; “Adapting the Proposed Method to Energy Efficient Clustering for WSN” section details the adaptation and application of the proposed method for clustering in WSN; “Simulations and Experimental Results” section presents the evaluation of experimental results, the comparison of the results to other algorithms, and a statistical analysis of the results, and finally, the “Conclusion” section summarizes and concludes the paper.

Basic Concepts

In this section, important information about the network model and the standard GSA are introduced and described.

The Network Model

In this paper, a network and radio model previously applied in [29] was used. The features of this network model are as follows:

- Sensor nodes sense information from the target area. Therefore, they always have information to send to the cluster head.
- The base station is a fixed node located outside the target area.
- The locations of all sensors are fixed.
- Each sensor has a limited battery.
- Sensor nodes can control and change the energy consumed in data transmission according to the distance of the destination node.
- All the sensor nodes can be selected as cluster heads.
- Sensed data are compressed in order to reduce the transmission information.

The radio model used in this paper is the first-order radio model proposed in [25]. The radio channel in this model is symmetric, meaning the energy consumption between the two-node transmissions is equal in two directions. The transmitter and receiver parts in each node consume energy in order to run the radio electronics and power amplifier. In this model, the energy consumption for data transmission between nodes i and j is dependent on the distance between these nodes, d_{ij}. The battery usage of the model is d_{ij}^2 for short distances, while the d_{ij}^4 model is employed for long distances. Hence, the total energy consumed in transmitting *k* bit data over a distance *d* is computed by the following equation:

$$E_T(k,d) = \begin{cases} k.E_{\text{el}} + k.\varepsilon_{\text{fs}}.d^2, \textit{if } d < d_0 \\ k.E_{\text{el}} + k.\varepsilon_{\text{tr}}.d^4, \textit{if } d \geq d_0 \end{cases}, \quad (1)$$

where E_{el} is the energy consumption for each bit to run the receiver or transmitter, ε_{fs} and ε_{tr} are dependent on the model of the transmitter amplifier utilized, and d_0 is a threshold for the transmitter distance. E_R, the energy expended for receiving *k* bit data, is computed as follows:

$$E_R(k) = k.E_{\text{el}}. \quad (2)$$

The energy consumption for receiving and compressing *k* bit data in the base station node is computed by the following equation:

$$E_{\text{BS}}(k) = E_R(k) + E_{\text{da}}, \quad (3)$$

where E_{da} is the energy cost for data aggregation and compression in the base station node. The parameters of this model which are used in the "Experimental and Simulation Results" section are defined as follows:

$$E_{\text{el}} = 50\frac{nJ}{\text{bit}}, \varepsilon_{\text{fs}} = 10\ \frac{pJ}{\text{bit}.m^2} \text{ and } \varepsilon_{\text{tr}} = 0.0013\frac{pJ}{\text{bit}.m^2}, \text{and } E_{\text{da}} = 5\frac{nJ}{\text{bit}}.$$

The Gravitational Search Algorithm

In the GSA, which is a stochastic and nature-inspired optimization algorithm, the agents are considered as objects while their masses represent their performance. The objects can transmit the information about multi-dimensional search space by their movements and gravitational forces. In GSA with *N* objects (agents), the position of i^{th} object at time *t* for $i = 1, \ldots N$ is denoted by:

$$X_i(t) = \left(x_i^1(t), \ldots, x_i^d(t), \ldots, x_i^m(t)\right) \quad i = 1, 2, \ldots, N, \quad (4)$$

where $x_i^d(t)$ shows the position of i^{th} agent at time *t* in the d^{th} dimension, and *m* refers to the search space dimension. The following equation calculates the overall amount of gravitational force based on the law of gravity acting on object *i*, in dimension *d*, at time *t*:

$$F_i^d(t) = \sum_{j\in K\text{best}(t), j\neq i} \textit{r}\text{and } G(t)\frac{M_i(t).M_j(t)}{R_{ij}(t)+\varepsilon}\left(x_j^d(t) - x_i^d(t)\right), \quad (5)$$

where $M_i(t)$ and $M_j(t)$ are the gravitational masses of the i^{th} and j^{th} objects at time *t*, respectively. *r*and is a uniform random variable within the interval [0, 1], and *G*(*t*), reflecting a decreasing function of time, is the gravitational constant in the t^{th} iteration. ε is another constant which is equal to a small value ($\varepsilon = 10^{-16}$). The objective of this constant is to avoid having the value of zero as the denominator of this fraction. *K*best(*t*) is the set of *K*(*t*) heavier objects at time *t* which is a decreasing function of time *t*, and $R_{ij}(t)$ denotes the Euclidean distance between objects *i* and *j*. *G*(*t*), *K*(*t*), and $R_{ij}(t)$ are computed as follows:

$$G(t) = G_0\exp\left(-\gamma\frac{t}{t_{\text{Max}}}\right), \quad (6)$$

$$K(t) = \left[\kappa + \left(1 - \frac{t}{t_{\max}}\right)(1-\kappa)\right] \times 100, \quad (7)$$

$$R_{ij}(t) = \left\| X_i(t), X_j(t) \right\|_2. \quad (8)$$

In the last iteration of GSA, only κ% of agents can apply gravitational force to the other individuals. In this paper, we consider $\kappa = 0.02$.

According to the Newtonian law of motion, the following equations can calculate the acceleration of object *i*, its velocity and new position in dimension *d*, at time *t*:

$$a_i^d(t) = \frac{F_i^d(t)}{M_i(t)}, \quad (9)$$

$$V_i^d(t+1) = \text{rand} \cdot V_i^d(t) + a_i^d(t), \quad (10)$$

$$X_i^d(t+1) = X_i^d(t) + V_i^d(t+1). \quad (11)$$

According to Eq. (5), heavier objects which are a better solution for the problem, are more effective agents. These agents have a greater absorption power where this power can attract other agents. Furthermore, the heavier agents move more slowly in the feasible area according to Eq. (9).

In GSA, the values of active, passive, and inertial masses are equal for the agent; this value for the i^{th} object is calculated using the corresponding normalized fitness as Eq. (12):

$$M_i(t) = \frac{\text{fit}_i(t) - \text{worst}(t)}{\sum_{j=1}^{N}\text{fit}_j(t) - \text{worst}(t)}, \quad (12)$$

where $\text{fit}_i(t)$ computes the fitness function for i^{th} agent in t^{th} iteration, while worst(*t*) in minimization problems is calculated as follows:

$$\text{worst}(t) = \max_{j\in\{1,\ldots,N\}}\text{fit}_j(t). \quad (13)$$

Rank-Based GSA

Search algorithms try to establish a good trade-off between exploitation and exploration power in order to find a global optimum solution for the problem [31]. The mass calculation in standard GSA is a very important issue to control and balance the abilities of this method, as the movement and acceleration of each agent in GSA are proportional to the value of masses of other agents (Eqs. (5) and (9)). In a standard GSA, the value of mass is computed according to Eq. (12), so that they are equal to the normalized fitness of agents. In ref. [22], the authors attempt to control the exploration and exploitation properties by defining new approaches to compute the value of masses. They use sigma scaling and Boltzmann selection functions to compute the value of the agent's masses and control the abilities of the GSA. However, in such methods, the value of masses is dependent on the fitness of agents. Therefore, in some applications and problems where their agent's fitness is substantially different and the standard deviation of normalized fitness is typically high, this method does not perform well carries the risk of premature convergence. A rank-based selection method, as used in this paper for mass calculation, can overcome these scaling problems. In this method, agents are sorted according to their fitness, and the values of their masses depend only on the relative fitness rather than on absolute fitness [32, 33].

Rank-Based Method

In a ranked based method for mass calculation, first, the agents are sorted according to the value of their fitness function. Therefore, the worst agent has Rank = 1, while the best agent has Rank = N, where N shows the number of agents. The value of the mass assigned to each agent depends only on its rank and position and not on the actual objective value [34]. Consider $R_i(t)$ as the rank of the i_{th} agent in the population at time t, and S as a selective pressure, which is the parameter of this method. The mass value for each agent in time t, $M_i(t)$, could then be calculated as follows [35]:

$$M_i(t) = (2-S) + \frac{2(S-1)(R_i(t)-1)}{N-1}. \tag{14}$$

In this method, parameter S which is allowed to be within [1.0, 2.0], can control the exploitation and exploration abilities of the algorithm. If S is close to 2 and obtains a large number, the standard deviation of the value of masses would be high, and major differences between the value of masses would be observed; therefore, the exploitation ability of the algorithm increases and the algorithm converges to the best solution. On the other hand, when the value of parameter S is low and close to 1, the standard deviation of the value of masses diminishes and the value of masses approach each together; in this case, the exploration ability of the algorithm would increase. The algorithm also searches the feasible area in order to find better solutions and avoid being trapped in local optima [13]. In a rank-based method, the value of parameter S is determined by the user; consequently, it is a user-dependent method. In this paper, the authors attempted to overcome this deficiency of rank-based methods using a FLC which can compute the value of parameter S automatically and independent of the user and the problem.

FLC to Control the Parameter of Rank-Based Method

In this section, the value of S, the parameter of the rank-based method, was computed using a fuzzy logic controller. The exploitation and exploration abilities of the algorithm were controlled using this FLC. The controller proposed for this method is called Periodic. This controller computes the parameter S and the rank-based GSA runs with this S for a period of time. The FLC updates the value of parameter S at some pre-defined update time with the algorithm employing this value during the period of time (p). To design a FLC, a fuzzy inference system was proposed which obtains three variables as its input and produces an output variable. IM(t), IT, and $S(t-p)$ are inputs of this controller, and $S(t)$ is the output of the FLC. IM(t) refers to the improvement of the algorithm within a period of time. Suppose that Best(t) and Best($t-p$) contain the best solution for the minimization problem in time t and $t-p$, respectively. Thus, the IM(t) would be computed as:

$$\text{IM}(t) = 1-\frac{\text{Best}(t)}{\text{Best}(t-p)}. \tag{15}$$

The value of this variable is allowed to lie within a [0, 1) interval. If the value of IM is low, the improvement of the algorithm would have been minor within the last period of time. Furthermore, the high value of this variable would imply that the improvement of the algorithm has been significant, and the algorithm has had a good performance in searching and finding the problem minimization in the last period of time.

IT is a variable which shows the remaining time of the algorithm. The value of this variable is computed as follows:

$$\text{IT} = \frac{t}{\text{max\_it}}, \tag{16}$$

where t is the time and max _ it refers to the maximum number of iterations in the algorithm. According to Eq. (16), the value of IT is allowed to lie within (0, 1]. The lower value of this variable suggests that it is at the beginning of the algorithm. On the other hand, the high value of this parameter indicates that it is close to the end of the algorithm. $S(t-p)$ and $S(t)$ are two variables representing the parameter of the rank-based method in the previous period of time and now. Figures 1, 2, and 3 display the membership functions for these variables used in the proposed algorithm.

Fig. 1 The membership function for IM

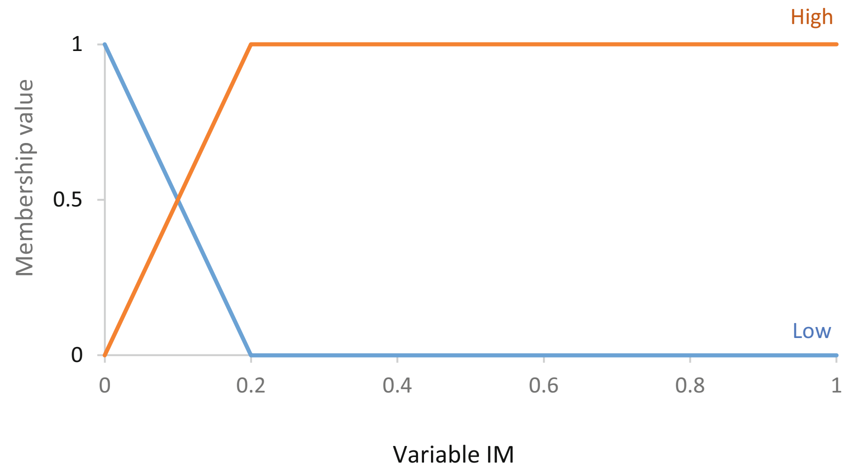


Table 1 presents the rules of this FLC. The proposed controller has three inputs such as those described in the previous section, so it should have $2 \times 2 \times 3 = 12$ rules as presented in Table 1. In this table, *L*, *M*, and *H* represent low, medium, and high levels for the corresponding variables. These rules are summoned to control the exploitation and exploration abilities of the algorithm.

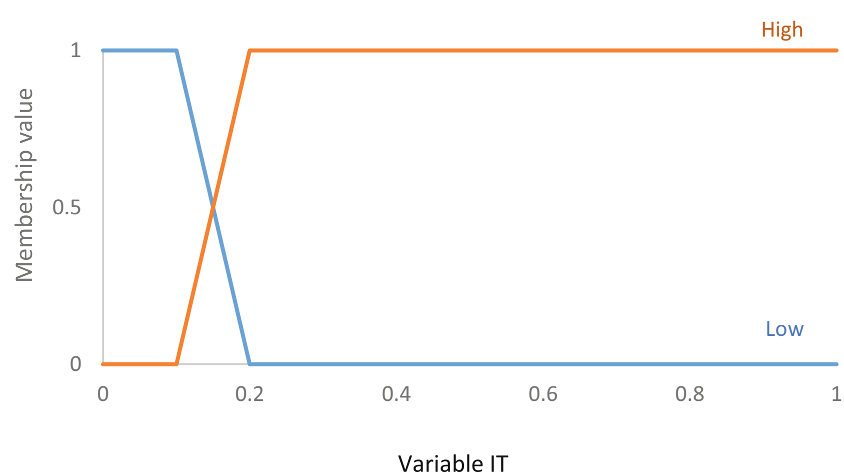
According to the second rule in Table 1, if there is a lack of improvement at the beginning of the algorithm, and the value of the selection pressure in the previous time period is medium, then the controller decreases the value of the rank-based parameter. The controller executes this task since the algorithm should explore the feasible area at the beginning of the algorithm to discover better solutions. Further, a low value of the variable IM(t) implies that the algorithm is trapped in a local optima. As mentioned in the previous section, the exploration power of the algorithm was enhanced while the value of S was decreased. Concerning the fourth rule, if the improvement of the algorithm is low, and the algorithm processes its last iterations, then the controller would raise the value of parameter S.

The input parameters were combined using the AND operator and the Center of Gravity (COG) method was used for defuzzification. COG provided a crisp value based on the center of gravity of the fuzzy set. The total area of the membership function distribution used to represent the combined control action was divided into a number of sub-areas. The area and the center of gravity of each sub-area was calculated and then the summation of all these sub-areas was taken to find the defuzzified value for a fuzzy set. The basic principle in the COG method is to find the point s^* where a vertical line would slice the aggregate into two equal masses. s^* was calculated as follows:

$$s^* = \frac{\int x.\mu_C(x)\mathrm{dx}}{\int \mu_C(x)\mathrm{dx}}, \tag{17}$$

where x is the support value of the inferred membership functions and $\mu_C(x)$ shows the maximum value of the inferred membership functions at a given support value. For example, consider the variables IM(t), IT, and $S(t-p)$ have the values 0.05, 0.2, and 1.3 at time t. According to the membership functions of Figs. 1, 2, and 3, these variables would have low, high, and medium fuzzy values respectively. Therefore, given the second rule of Table 1, the FLC would increase the value of the variable $S(t)$. The proposed FLC calculated the

Fig. 2 The membership function for IT



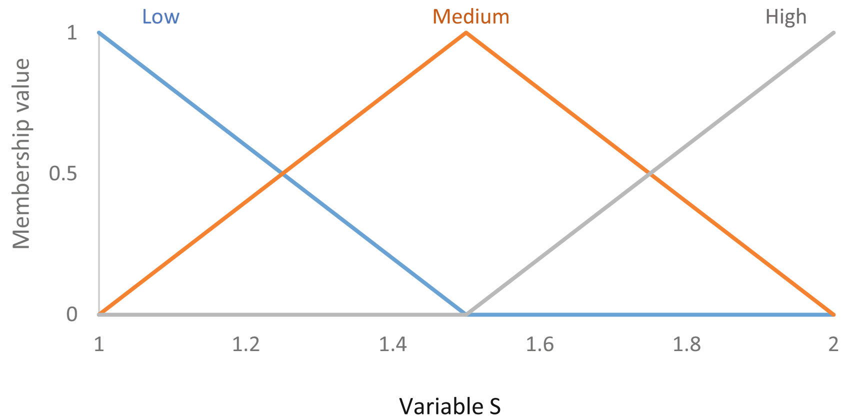


Fig. 3 The membership function for S

value of variable $S(t)$ using Eq. (17) as $(t) = 1.576$.

Adapting the Proposed Method to Energy Efficient Clustering for WSN

The WSN clustering protocol used in this paper contained multiple rounds. Each round consisted of two parts: a setup phase at the beginning of each round, and a steady state phase in the continuous round. At the beginning of each round, the algorithm arranged the clusters and defined the cluster heads. Then, in the steady state part, the sensors sensed information from the target area and relayed it to their corresponding cluster heads. Cluster heads then compressed the information and transmitted it to the base station.

The calculations of the setup part were done by the base station, which is a node with a high supply battery unit. Nodes sent their properties and local information such as residual energy and their position to the base station. This node calculated the average residual energy of total sensors and specified cluster head candidates which were nodes and whose remaining energy was greater than average. Then, an optimization algorithm ran on the predefined cost function to determine cluster heads among these cluster head candidates. The cost function was defined as follows [29]:

$$\text{cost} = \alpha \times f_1 + (1-\alpha) \times f_{2,} \tag{18}$$

$$f_1 = \max_{h=1,\dots,H}\left\{\sum\nolimits_{\forall n_i \in C_{p,h}} d\left(n_i, CH_{p,h}\right)/|C_{p,h}|\right\}, \tag{19}$$

$$f_2 = \sum\nolimits_{i=1}^{N} E(n_i)/\sum\nolimits_{h=1}^{H} E\left(CH_{p,h}\right). \tag{20}$$

As demonstrated earlier, this cost function is a combination of two functions. The first function, f_1, minimizes the intra distance in the clusters while the second one, f_2, enhances the total remaining energy of the network. In these equations, α can control the effect of each cost function; H is the number of clusters; $d(n_i, \mathrm{CH}_{p,\ h})$ indicates the Euclidian distance between node n_i and cluster heads $\mathrm{CH}_{p,\ h}$, and $E(n_i)$ compute the remaining energy of node i. $|C_{p,\ h}|$ denotes the number of nodes belonging to cluster C_h in the p^{th} optimization algorithm agent. The optimization algorithm finds the best agent in order to minimize the cost function. This agent determines cluster heads and organizes clusters in the WSN such that there are compact clusters with high remaining energy corresponding cluster heads.

The final objective function of the optimization problem proposed in this paper and based on the above defined cost function was defined as the following equation:

$$\text{Objective function} = \text{Argmin}_{\mathrm{CH}_i}\ \text{cost}. \tag{21}$$

The positions of all cluster heads constitute the variables in this objective function. In this paper, a 2D area was considered as the target space; hence, two elements were required to show the position of each cluster head in the feasible space. Therefore, each solution for this objective function contained 2H variables, where H was the number of cluster heads.

In this paper, the improved version of GSA was used to solve this optimization problem while attempting to find the best answer for this problem in order to minimize the objective function. The best solution organized the compact clusters whose cluster heads had a high remaining energy. In the proposed GSA, each agent was a solution for the objective

Table 1 Fuzzy rules for the proposed FLC

Rule		IM(t)	IT	S (t-p)		S (t)
1		L	L	L		L
2		L	L	M		L
3		L	L	H		M
4		L	H	L		M
5		L	H	M		H
6	If	L	H	H	Then	H
7		H	L	L		L
8		H	L	M		M
9		H	L	H		H
10		H	H	L		L
11		H	H	M		M
12		H	H	H		H

function. It contained 2H real values determining the locations of H cluster heads. Figure 4 reveals the structure of each agent in the modified GSA version. The fitness value of each agent was computed using Eq. (18). The variables in the algorithm were updated with the FLC calculating the selection pressure. The value of masses was computed using a rank-based method, where agents moved towards their new location and the next iteration was executed. Figure 5. demonstrates the flowchart of computing and determining cluster heads using the proposed GSA algorithm.

In this simulation, sensor nodes had fixed and predetermined locations in the target area. Thus, the variables in the fitness function (the location of each cluster head) could not obtain any value. Therefore, the map function was used to map the value of the real answer to the nearest sensor location as a feasible solution for the problem.

Figure 5. demonstrates the flowchart of computing and determining cluster heads using the proposed GSA algorithm.

The base station selected cluster heads and organized clusters in WSN using the proposed algorithm. It sent information and cluster head IDs to all nodes in the network and cluster heads organized a time-division multiple access (TDMA) schedule in order to avoid data collisions. Further, each sensor node could turn its radio device on when it communicated with other nodes and turn it off otherwise. Therefore, TDMA schedule could potentially reduce energy consumption in each sensor node. The sensed information was transmitted to the cluster heads. Cluster heads compressed the received information and achieved data fusion. Finally, they transferred their compacted data to the base station.

Simulations and Experimental Results

For any new optimization method, it is essential to validate its performance and compare it with other existing algorithms over a good set of benchmark and test functions. This large set of test functions should contain a wide variety of test functions, such as unimodal, multimodal, and composition problems.

In this section, first, the performance of the modified GSA was compared with that of the most commonly used BIAs and meta-heuristic algorithms. The proposed model was evaluated on two sets of standard benchmark functions. Then, the algorithm was used in the WSN clustering method as demonstrated in "Adapting the Proposed Method to Energy Efficient Clustering for WSN" section. The performance of this method was also evaluated and compared to that of other energy efficient clustering protocols for WSNs. After obtaining the experimental results, the statistical tests were used for further analysis and to compare the performance of the proposed method to that of other algorithms. In this paper, Friedman's method [36], Friedman aligned ranks test [37], and binomial sign test [38] which are non-parametric statistical test methods, were used to analyze the results.

Experimental Results on the CEC Functions

First, the behavior of the proposed algorithm, rank-based GSA (RB-GSA), was compared to that of some popular biologically inspired optimization algorithms such as Joint Approximation Diagonalization of Eigen-matrices (JADEEP) [39], Global PSO (G-PSO) [40], RGA [41], standard GSA [13], Distribution-GSA [42], Black-Hole-GSA [17], and Clustered-GSA [16] on the CEC2013, the real-parameter single objective optimization benchmark which is a common standard benchmark or test function [43]. This benchmark set consisted of 28 scalable test functions where the global optimum was shifted to o, $o = (o_1, \ldots, o_D)$ (D is the number of dimensions). In this set, functions $F_1 - F_5$ were unimodal; F_6-F_{20} were multimodal; and F_{17} and F_{18} were double-funnel functions with a high deceptive structure. Finally, $F_{21} - F_{28}$ were composite functions which combined multiple test problems into a complex landscape.

All these minimization problems and test functions had the global optimum, which is equal to zero, in their definition bounds. The result error values of these algorithms on CEC 2013 standard test functions are presented in Table 3. This table contains the average of the best answers which were computed on 51 independent runs after 1e+5 Fitness Evaluations (FE) with a dimension of 50 ($n = 50$) via these algorithms [44]. In this table, the italic numbers indicate the best solution of mean values for each test function.

In all kinds of GSA algorithms (GSA, D-GSA, BH-GSA, and C-GSA), all the parameters were set as described in [13]. In these algorithms, the total number of agents, N, was set to 50, and the maximum number of iterations was set to 2000 ($t_{\mathrm{Max}} = 2000$).

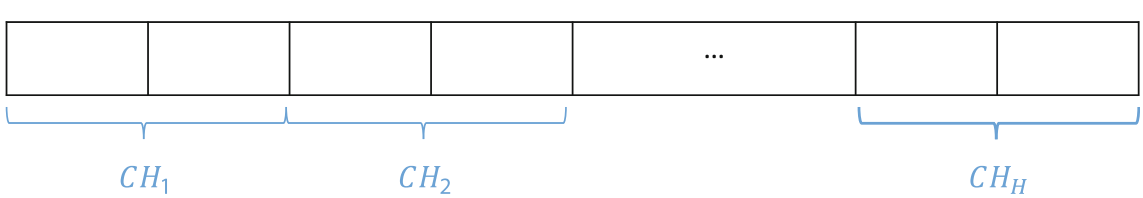


Fig. 4 The structure of each agent in the rank-based GSA which is used to solve this optimization problem

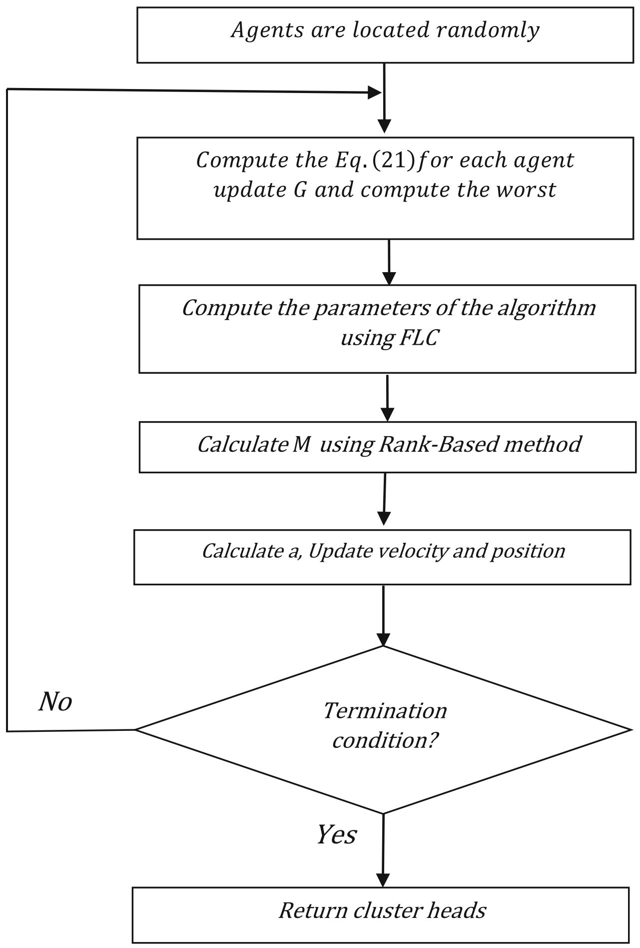


Fig. 5 Adapted proposed algorithm for determining cluster heads in the setup phase

The parameters of BH-GSA and CGSA were set as described in [16, 17] respectively. Table 2 summarizes the parameter settings of these optimization approaches.

Table 3 reveals that the proposed algorithm had a good performance for test functions F_1, F_8, F_{12}, F_{13}, F_{16}, F_{19}, and F_{22}. JADEEP found better solutions for functions F_1, F_2, F_4, F_5, F_{11}, F_{14}, F_{21}, and F_{28}. Also, BH-GSA had a good performance on functions F_3, F_7, F_8, F_9, F_{10}, F_{15}, F_{24}, F_{25}, F_{26}, and F_{27}. BH-GSA found a good solution for the composition functions. Nevertheless, the results of RB-GSA were close to those of BH-GSA for these kinds of test functions. GSA had a better performance for functions F_1, F_8, F_{17}, F_{18}, and F_{23}, while D-GSA and C-GSA had a good performance for functions F_8, F_{20}, F_{23} and F_6, F_{17}, F_{19} respectively.

Table 2 Parameter settings of the optimization algorithms

Algorithms	Parameters setting
RGA	Pc = *0.3,* Pm = *0.1*
GSO	$\varphi 0=\pi/4, a=\text{round}(\sqrt{n+1}), \theta_{\max}=\pi/a^2$, percentageOfRangers = 20%
MWO	$\alpha=1.1$, $\beta=7.5$, $a=0.63$, $b=1.26$, $c=1.05$, $\gamma=0.1$
GPSO and PSO	$\omega=0.9-0.4$, $c1=c2=2$
JADEEP	$1/c\in[5, 20]$, $p\in[5\%, 20\%]$
GSA	$G0=100$, $\gamma=20$
D-GSA	$G0=100$, $\gamma=20$, $\theta=100$, $\rho=10^{-16}$
BH-GSA	$G0=100$, $\gamma=20$
C-GSA	$G0=100$, $\gamma=20$, ClusterIt = 1510, ClusterStep = 10
RB-GSA	$G0=100$, $\gamma=20$

In this section, Friedman test was used to compare and to assign ranking to the performance of different algorithms. Further, binomial sign test was used for pairwise comparison between the proposed method and other approaches. These methods are non-parametric statistical tests and do not make assumption about the parameters of the population distribution of the data.

In these methods, the significance level was set to 0.1, where the null-hypothesis was that the average performances of the algorithms are equal while the alternative hypothesis was that the average performances of the algorithms are not equal. r_i^j was considered as the rank of the j^{th} of k methods, on the i^{th} of n benchmark functions such that the ranking 1 was assigned to the best of them and the ranking k to the worst. The Friedman test needs the computation of the average ranks of different approaches, $R_j=\frac{1}{n}\sum_i r_i^j$. Under the null hypothesis, the Friedman statistic were calculated with the following equation:

$$\chi_F^2=\frac{12n}{k(k+1)}\left[\sum_j R_j^2-\frac{k(k+1)^2}{4}\right]. \tag{22}$$

This statistic is distributed according to χ_F^2 with $k-1$ degrees of freedom, where n and k are big enough. [36, 38].

The STAC platform [45] was used to conduct this statistical test. Table 4 provides the results of the Friedman test for the minimization results of Table 3. The average rank values in Table 4 indicate that the proposed method claims the first rank has the best performance compared to other algorithms. The last row of Table 4 shows the statistics and p value of the test. The null-hypothesis was rejected, and the results show that the RB-GSA outperformed the state-of-the-art optimization algorithms by controlling the exploitation and exploration abilities.

Table 5 reports the results of the binomial sign test on the CEC functions. In this table, each row compares the performance of the proposed optimization method with that of other methods and analyzes the experimental results. According to the results of Tables 4 and 5, there is a significant difference between the performances of the proposed algorithm, the best-ranked method, and other approaches.

Figures 6 and 7 compare the performance of the proposed method and standard GSA on functions F_3 and F_{24}. These

Table 3 Minimization results on CEC test functions

GPO	JADEEP	GSA	D-GSA	BH-GSA	C-GSA	RB-GSA	
F_1	5.29E4.03	*0.00E+00*	*0.00E+00*	4.50E−01	5.35E−14	1.42E−12	*0.00E+00*
F_2	8.34E+07	*5.43E+03*	1.79E+06	6.78E+06	1.73E+06	2.09E+06	6.83E+03
F_3	9.81E+10	6.19E+06	1.32E+08	4.97E+08	*1.39E+05*	1.37E+08	4.28E+05
F_4	1.58E+04	*5.00E+03*	1.81E+04	1.70E+04	1.49E+04	1.88E+04	1.72E+04
F_5	1.59E+03	*0.00E+00*	5.61E−05	2.79E+01	7.09E−05	5.97E−05	7.20E−06
F_6	4.88E+02	9.09E−01	5.03E+01	6.90E+01	7.09E−05	*5.97E-05*	6.14E−04
F_7	1.63E+02	4.67E+00	1.97E+01	2.86E+01	*1.23E+00*	1.83E+01	7.81E+00
F_8	2.12E+01	2.09E+01	*2.04E+01*	*2.04E+01*	*2.04E+01*	2.10E+01	*2.04E+01*
F_9	4.43E+01	2.69E+01	4.23E+00	4.71E+00	*1.63E+00*	4.16E+00	2.17E+00
F_{10}	1.45E+03	3.75E−02	1.19E−02	1.38E+00	*7.25E−04*	6.62E−03	9.66E−04
F_{11}	1.56E+02	*0.00E+00*	2.55E+01	2.60E+01	5.17E+00	2.60E+01	2.22E+01
F_{12}	3.73E+02	2.06E+01	2.36E+01	2.45E+01	3.51E+01	2.30E+01	*1.77E+01*
F_{13}	5.87E+02	4.16E+01	4.80E+01	4.47E+01	6.57E+01	4.70E+01	*3.95E+01*
F_{14}	2.59E+03	*4.39E−02*	8.62E+02	8.42E+02	3.13E+02	8.82E+02	4.33E+01
F_{15}	7.76E+03	3.20E+03	5.12E+02	4.56E+02	*2.93E+02*	4.89E+02	3.17E+02
F_{16}	2.09E+00	1.75E+00	1.52E−02	1.14E+00	2.07E−02	1.49E−02	*9.01E−03*
F_{17}	3.46E+02	3.04E+01	*1.30E+01*	2.70E+01	1.32E+01	*1.30E+01*	1.34E+01
F_{18}	3.45E+02	7.31E+01	*1.30E+01*	3.53E+01	1.38E+01	1.35E+01	1.34E+01
F_{19}	4.36E+04	1.43E+00	1.28E+00	1.63E+00	1.37E+00	*1.14E+00*	*1.14E+00*
F_{20}	2.22E+01	1.01E+01	4.26E+00	*4.00E+00*	4.02E+00	4.70E+00	4.96E+00
F_{21}	9.33E+02	*2.98E+02*	4.00E+02	4.00E+02	4.00.E+02	4.00E+02	4.00E+02
F_{22}	4.25E+03	1.93E+03	2.02E+03	2.02E+03	4.22E+03	1.98E+03	*1.86E+03*
F_{23}	1.06E+04	3.25E+03	*1.28E+03*	*1.28E+03*	4.63E+03	1.33E+03	1.29E+03
F_{24}	3.37E+02	2.10E+02	2.26E+02	2.27E+02	*2.07E+02*	2.24E+02	2.21E+02
F_{25}	4.81E+02	2.63E+02	2.13E+02	2.15E+02	*2.01E+02*	2.15E+02	2.09E+02
F_{26}	4.17E+02	2.09E+02	3.16E+02	2.83E+02	*1.42E+02*	3.82E+02	3.05E+02
F_{27}	1.68E+03	5.34E+02	4.00E+02	4.01E+02	*3.59E+02*	4.00E+02	3.96E+02
F_{28}	4.33E+03	*3.00E+02*	6.52E+02	6.67E+02	3.14E+02	6.38E+02	3.74E+02

figures reveal that the performance of the GSA was improved by using a rank-based method for mass calculation and by using the controller to define the parameters of the algorithm. In the proposed method, exploitation and exploration abilities of the algorithm can be controlled using the FLC. Thus, the RB-GSA can enhance the exploration ability if it is required to search the feasible area to find a better solution. Further, the RB-GSA algorithm can control its exploitation power. The proposed algorithm can also avoid trapping in local optima by controlling and balancing its abilities.

As these figures illustrate, the rank-based method improves the performance of GSA. Function F_3 is a unimodal function in which the rank-based method can improve the exploration ability of the algorithm in order to explore the feasible area for finding

Table 4 Results of Freidman test on the CEC functions

Algorithms	Rank
RB-GSA	2.6428
BH-GSA	2.8035
JADEEP	3.3928
C-GSA	3.7678
GSA	3.9107
D-GSA	4.6250
G-PSO	6.8571
Statistic	*p* value
21.0233	le-5 ≡ rejected

Table 5 Results of binomial sign test on the CEC functions

Methods	Statistic	*p* value	Hypothesis
RB-GS vs. BH-GSA	18	0.0969	Rejected
RB-GS vs. JADEEP	19	0.0820	Rejected
RB-GS vs. C-GSA	23	le-5	Rejected
RB-GS vs. GSA	21	0.0001	Rejected
RB-GS vs. GSA	22	8e-5	Rejected
RB-GS vs. G-PSO	27	7e-9	Rejected

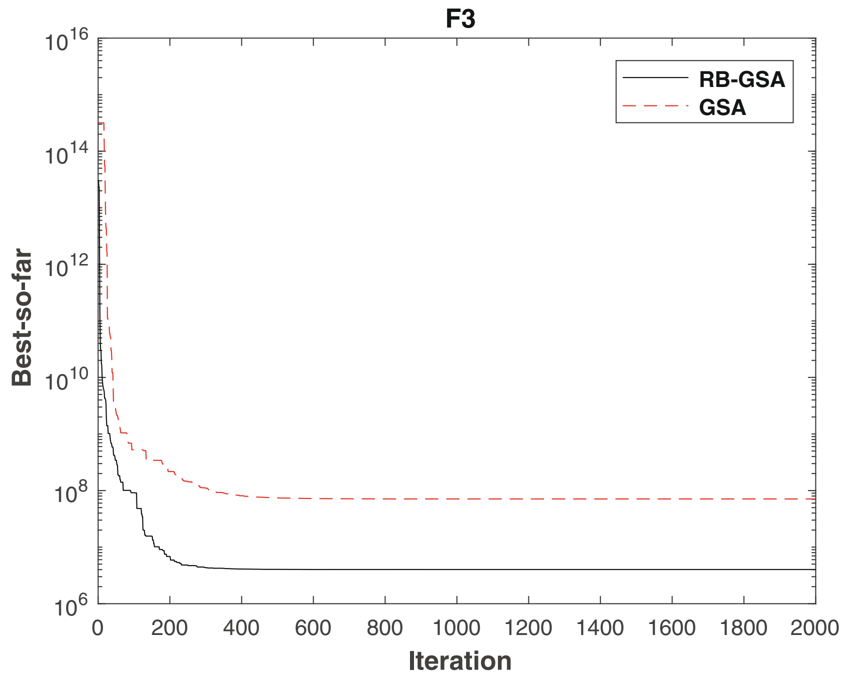


Fig. 6 Comparison of the performance of the proposed method and GSA for minimization of F_3

better solutions for the problem. On the other hand, function F_{24} is a multimodal function in which there are many local optima. The proposed method balances the exploitation and exploration abilities to avoid being trapped in these local optima.

According to the experimental results, the modified version of GSA has a good performance for multi-modal functions which are sophisticated and complicated functions with many local optima. Rank-based method can control the exploitation and exploration of the algorithm using a fuzzy logic controller in order to avoid being trapped in local optima on these functions. Although the proposed method has greater time complexity against standard GSA, as illustrated in the "Rank-Based GSA" section, it can find more accurate solutions for complicated problems.

Experimental Results on the Standard Benchmark Functions

In this section, the performance of rank-based GSA on eight standard benchmark functions previously introduced and used in [46] were evaluated. These test functions have been widely used in the literature to compare the performance of different

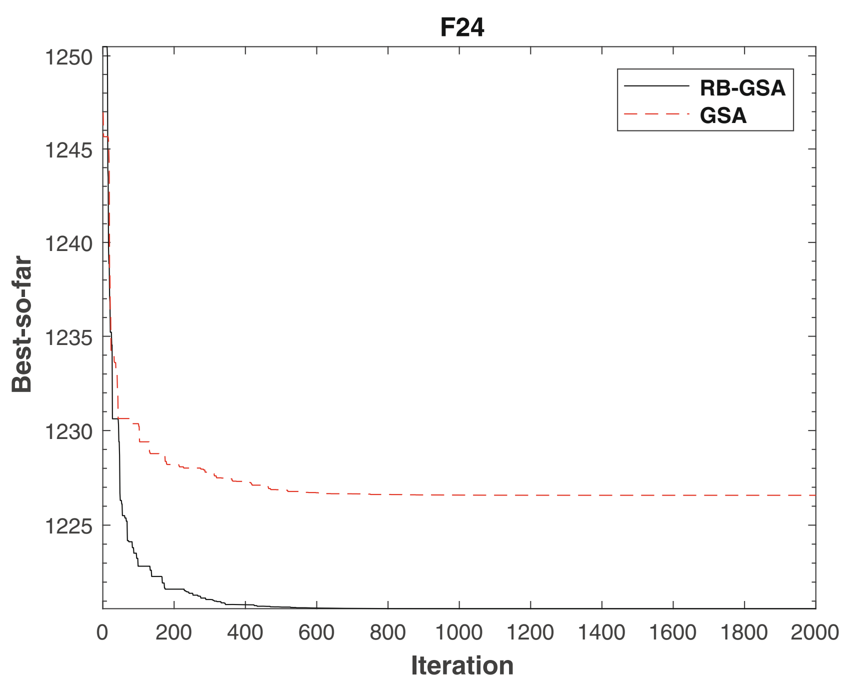


Fig. 7 Comparison of the performance of the proposed method and GSA for minimization of F_{24}

Table 6 Minimization results on the standard test functions

	RGA	PSO	GSO	MWO	GSA	RB-GSA
F_1	138.81	0.59	157.11	3.87e-6	4.08e-11	*7.36e-l2*
F_2	4.19e+4	4.54e+5	3.59e+4	142.68	120.20	*112.26*
F_3	1.26e-4	5.20e-3	9.87e-2	*1.02e-17*	9.66e-4	9.66e-4
F_4	64.18	43.62	91.23	*17.09*	25.50	22.28
F_5	2.73	0.18	1.17	2.96e-2	1.19e-2	*9.66e-4*
F_6	11.70	6.48	*3.34e-3*	6.61e-2	3.16	2.04
F_7	8.62	63.71	17.50	2.32e-2	*2.18e-2*	4.86e-2
F_8	0.57	24.18	1.90	1.95e-2	5.62e-4	*6.36e-4*

optimization algorithms. The performance of the proposed method was compared with that of other well-known optimization algorithms such as RGA [35], PSO [47], group search optimizer (GSO) [48], mussels wandering optimization (MWO) [46], and standard GSA algorithm [13].

In this paper, 50 independent runs were executed for each approach on every test function, with the average of these simulations being reported in Table 5. The dimension of each function is set to 30 and the maximum number of iterations is set to 1000. The experimental results were obtained through a PC with a 2.6GHz Intel Core i7-6700HQ processor and 16 GB RAM. The operating system was Windows 10 and all the codes were written and executed in MATLAB.

Table 6 compares the performance of different optimization methods on the standard test functions defined in [46]. In this table, the best solution for each function is italic. It can be seen that the proposed method has the best performance on functions F_1, F_2, F_5, and F_8.

Table 7 presents the results of Friedman test for minimization results of the standard test functions defined in [46]. The average rank values in Table 7 indicate the proposed method to have the best performance compared to other algorithms on these test functions. The statistic and the p value of this test are shown in the last row. The significance level was set to 0.1 and the null-hypothesis was rejected in this test. Table 8 shows the results of the pairwise comparison binomial sign test. In this table, the performance of the proposed method is compared with other approaches. The rows in this table test the hypothesis in which the significance value was set to 0.1. The results of Table 4 and Table 7 reveal that using a rank-based approach to control the abilities of the GSA can improve the performance of the optimization algorithm in many test functions. On the other hand, the results of Table 8 show that there is a significant difference between the performance of the proposed method and other optimization algorithms on these benchmark functions. Accordingly, it can be a good choice for solving optimization problems in real-world applications.

Table 7 Results of Freidman test on the test functions

Algorithms	Rank
RB-GSA	2.0625
MWO	2.2500
GSA	2.3125
RGA	4.6250
GSO	4.7500
PSO	5.0000
Statistic	p value
9.5925	le-5 ≡ rejected

Table 8 Results of binomial sign test on the test functions

Methods	Statistic	p value	Hypothesis
RB-GSA vs. MWO	6	0.0703	Rejected
RB-GSA vs. GSA	8	0	Rejected
RB-GSA vs. RGA	8	0	Rejected
RB-GSA vs. GSO	7	0.0078	Rejected
RB-GSA vs. PSO	8	0	Rejected

The Results of Clustering Methods

The behavior of the suggested clustering method introduced and described in the previous section is evaluated and compared with other clustering protocols for WSNs using MATLAB simulator. One hundred nodes are broadcasted randomly in the 100 m × 100 m target area. The initial energy for each of these nodes is 0.5 J and the number of clusters, H, is set to 10. In this paper, two scenarios were considered given the location of the base station. In the first one, the base station was located at (50, 175), which was outside of the target area and far from other sensors. In the second scenario, the base station was located at (50, 50) which was the center of the network area. The packet length sent from nodes to their corresponding cluster head was 200 bits, while the packet length of messages sent from cluster heads to the base station was considered as 6400 bits.

The performance of the proposed protocol, RB-GSA-C, which employs a rank-based GSA to find the best cluster

Table 9 Parameter settings of the clustering algorithms

Algorithms	Parameters setting
EECA	$Rs=10$, $Rc=20$
CSA	Max annealing temperature = 1000
E-OEERP	$R=30m$, Sensing range $=36m$, $\omega=[0.9–0.4]$, cl $=c2=2$
PSO-C	$\omega=[0.9-0.4]$, cl $=c2=2$, $\beta=0.5$
RB-GSA	$G0=100$, $\gamma=20$, $\alpha=0.5$

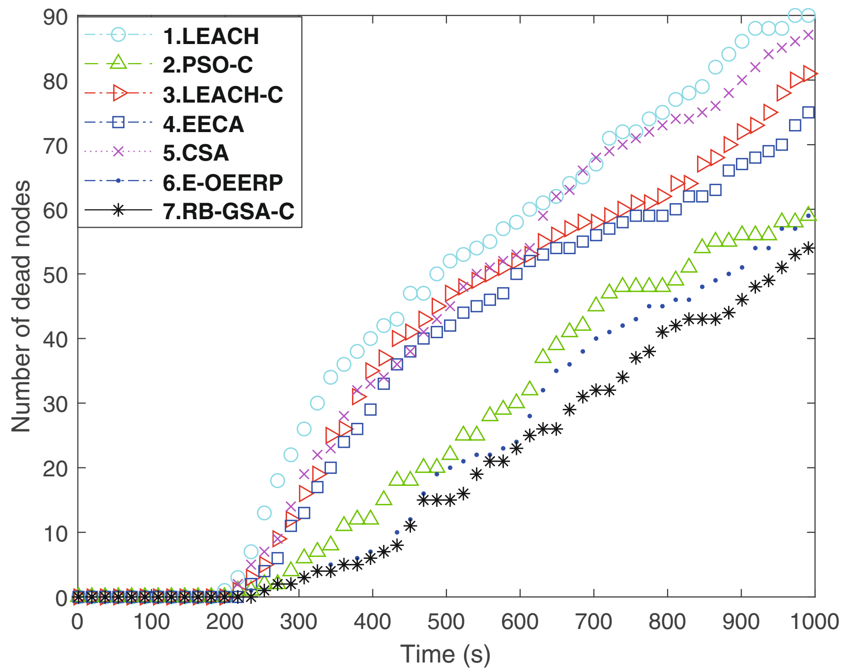


Fig. 8 Number of dead nodes over time with base station location at (50, 175)

heads, was compared with that of other popular clustering protocols for WSNs such as LEACH [25], LEACH-C [26], energy efficient clustering algorithm (EECA) [49], PSO-clustering method [29], cluster optimization based on simulated annealing (CSA) [50], and Enhanced Optimized Energy Efficient Routing Protocol (E-OEERP) [51]. This simulation ran until all the remaining energy of the sensors was depleted. The network model and its parameters described in the "Basic Concepts" section were used in this simulation. The parameters of GSA algorithm were set similar to [13]. The maximum annealing temperature in CSA was set to 1000. The parameter p in LEACH was set to 0.1, and the parameters of PSO algorithm were set similar to Ref. [52]. Table 9 summarizes the parameter settings of these clustering algorithms.

Figures 8 and 9 illustrate the simulation results of the first scenario where the base station was located outside the target area. Figure 8 compares the lifetime of these clustering algorithms. In this figure, the performance of these protocols was compared in terms of the number of dead nodes over time in WSNs. The proposed method, RB-GSA-C, outperformed other clustering methods, and WSNs employing this protocol for clustering showed a long lifetime.

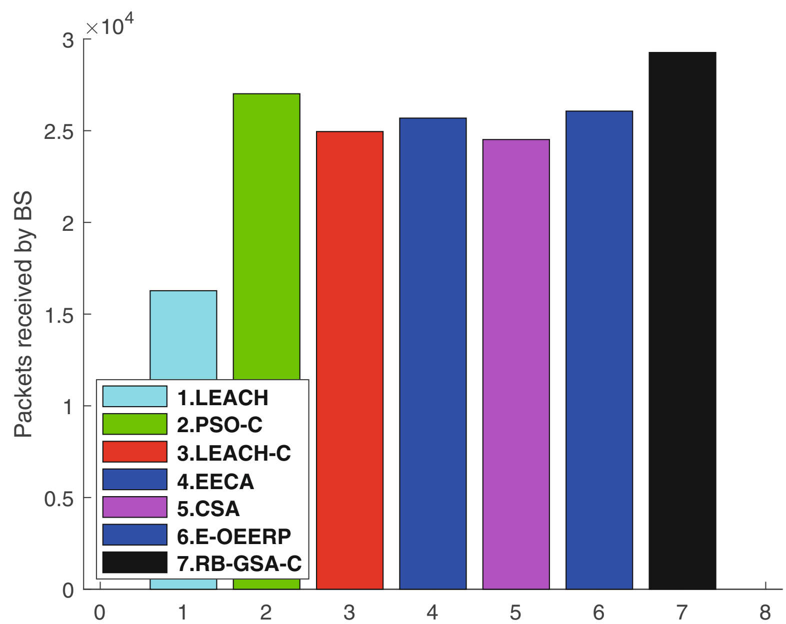


Fig. 9 Total data packets received by BS with base station location at (50, 175)

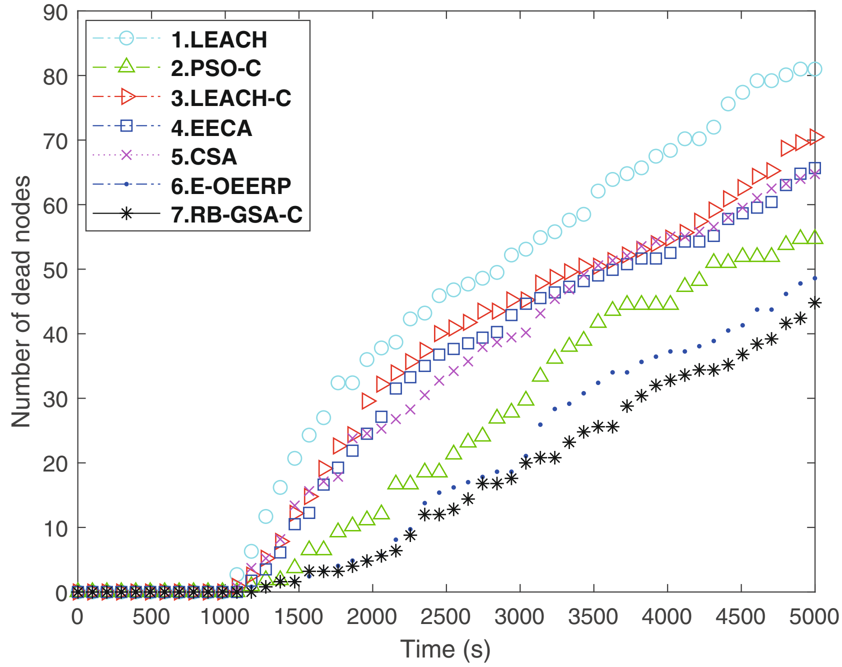


Fig. 10 Number of dead nodes over time with base station location at (50, 50)

Figure 9 compares the performance of clustering protocols in terms of the number of packets received by the base station. This figure reveals the effectiveness of the proposed clustering protocol in delivering more data messages to the base station.

Figures 10 and 11 display the simulation results of the second scenario where the base station is located inside the target area. Figure 10 compares the lifetime of these clustering algorithms. In this figure, the performance of these protocols is compared in terms of the number of dead nodes over time in WSNs. On the other hand, Fig. 11 compares the performance of clustering protocols in terms of the number of packets received by the base station.

When the number of algorithms for comparison is small, comparability among benchmark functions is desirable. This can be done with the aligned ranks method [37].

Table 10 presents the results of the Friedman aligned ranks test on the WSN clustering problem and compares the performance of different algorithms on this application problem. This method is used when the number of samples is less than algorithms. The rank values aligned in this table indicate that the proposed clustering method has the best performance

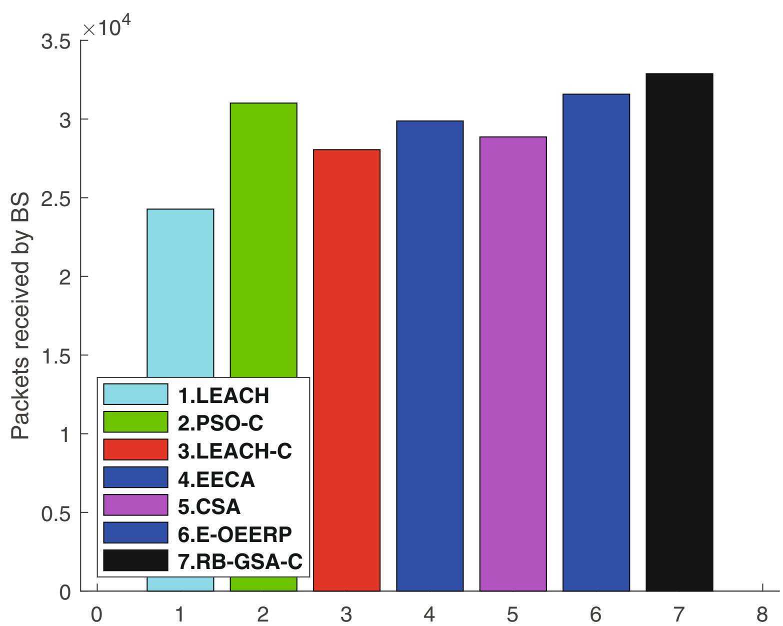


Fig. 11 Total data packets received by BS with base station location at (50, 50)

Table 10 Results of Freidman test on the WSN clustering problem

Algorithms	Rank
RB-GSA-C	6.50
PSO-C	8.50
E-OEERP	10.50
EECA	12.50
LEACH-C	14.50
CSA	23.50
LEACH	25.50
Statistic	p value
19.6266	0.0032 ≡ rejected

among the clustering algorithms. The statistic and the p value of this test are shown in the last row of Table 10. The null-hypothesis is rejected in this test at a 10% significance level. The significant performance achievement of this protocol is due to the application of the modified GSA version. As presented in the "Rank-Based GSA" and "Adapting the Proposed Method to Energy Efficient Clustering for WSN" sections, rank-based selections can control the performance of the optimization algorithm. The exploitation and exploration abilities in the modified GSA are controlled by the rank-based selection scheme and the FLC. Thus, RB-GSA has displayed a better performance in most optimization problems and the use of this algorithm has improved the performance of the energy efficient clustering method for WSNs.

Conclusion

In this paper, a rank-based fitness assignment method was used to calculate the value of masses in the GSA. This method was applied to overcome the scaling problems of the proportional fitness assignment. In the proposed algorithm, values of agent masses were dependent only on relative fitness rather than on the absolute objective functions. Controlling values of parameters was an effective solution to improve the optimization performance of a meta-heuristic algorithm by automatically setting the parameters to appropriate values during computational processes of the evolutionary search. Further, in this paper, a FLC was proposed to calculate and compute the parameter of rank-based method automatically and independent of the user. The exploitation and exploration abilities of this algorithm could be controlled by this FLC. The performance of the proposed optimization method was assessed by simulation and compared against other popular and state-of-the-art BIAs and optimization methods. The simulation results and statistical assessment revealed the proposed method to outperform the other methods. The difference between the performance of the proposed method and that of other approaches was a significant one. Therefore, it can be said that the proposed algorithm is an effective method to improve results while also having simplicity in implementation.

Then, the new version of GSA was applied to energy efficient clustering protocols for WSNs in order to optimize the objective function. This objective function defined compact clusters with more remaining energy nodes as their cluster heads. The observed experiments and comparisons suggested that, by using RB-GSA, the proposed approach outperformed other popular energy efficient clustering methods for WSNs.

Compliance with Ethical Standards

Conflict of Interest The authors declare that they have no conflict of interest.

Informed Consent Informed consent was not required as no human or animals were involved.

Human and Animal Rights This article does not contain any studies with human or animal subjects performed by any of the authors.

References

1. Tang W, Wu Q. Biologically inspired optimization: a review. Trans Inst Meas Control. 2009;31(6):495–515.
2. Molina D, LaTorre A, Herrera F. An insight into bio-inspired and evolutionary algorithms for global optimization: review, analysis, and lessons learnt over a decade of competitions. Cogn Comput. 2018;10:517–44.
3. Al-Rifaie MM, Bishop JM, Caines S. Creativity and autonomy in swarm intelligence systems. Cogn Comput. 2012;4(3):320–31.
4. Bishop JM, Erden YJ. Computational creativity, intelligence and autonomy. Cogn Comput. 2012;4(3):209–11.
5. Song B, Wang Z, Zou L. On global smooth path planning for mobile robots using a novel multimodal delayed PSO algorithm. Cogn Comput. 2017;9(1):5–17.
6. Kim S-S, McLoone S, Byeon JH, Lee S, Liu H. Cognitively inspired artificial bee colony clustering for cognitive wireless sensor networks. Cogn Comput. 2017;9(2):207–24.
7. Tang Q, Shen Y, Hu C, Zeng J, Gong W. Swarm intelligence: based cooperation optimization of multi-modal functions. Cogn Comput. 2013;5(1):48–55.
8. Siddique N, Adeli H. Nature-inspired chemical reaction optimisation algorithms. Cogn Comput. 2017;9(4):411–22.
9. Chakraborty S, Dey N, Samanta S, Ashour AS, Barna C, Balas MM. Optimization of non-rigid demons registration using cuckoo search algorithm. Cogn Comput. 2017;9(6):817–26.
10. Zhang A et al. Clustering of remote sensing imagery using a social recognition-based multi-objective gravitational search algorithm. Cogn Comput, 2018: 1–10.
11. Nisar S et al. Cognitively inspired feature extraction and speech recognition for automated hearing loss testing. Cogn Comput, 2019: 1–14.
12. Ghanem WA, Jantan A. A cognitively inspired hybridization of artificial bee colony and dragonfly algorithms for training multi-layer perceptrons. Cogn Comput. 2018;10(6):1096–134.
13. Rashedi E, Nezamabadi-Pour H, Saryazdi S. GSA: a gravitational search algorithm. Inf Sci. 2009;179(13):2232–48.
14. Rashedi E, Nezamabadi-Pour H, Saryazdi S. BGSA: binary gravitational search algorithm. Nat Comput. 2010;9(3):727–45.

15. Rashedi E, Rashedi E, Nezamabadi-pour H. A comprehensive survey on gravitational search algorithm. Swarm and evolutionary computation, 2018
16. Shams M, Rashedi E, Hakimi A. Clustered-gravitational search algorithm and its application in parameter optimization of a low noise amplifier. Appl Math Comput. 2015;258:436–53.
17. Doraghinejad M, Nezamabadi-pour H. Black hole: a new operator for gravitational search algorithm. Int J Comput Intell Syst. 2014;7(5):809–26.
18. Kherabadi HA, Mood SE, Javidi MM. Mutation: a new operator in gravitational search algorithm using fuzzy controller. Cybernet Inform Technol. 2017;17(1):72–86.
19. Valdez F, Melin P, Castillo O. A survey on nature-inspired optimization algorithms with fuzzy logic for dynamic parameter adaptation. Expert Syst Appl. 2014;41(14):6459–66.
20. Valdez F, Melin P, Castillo O. An improved evolutionary method with fuzzy logic for combining particle swarm optimization and genetic algorithms. Appl Soft Comput. 2011;11(2):2625–32.
21. Chang B-M, Tsai H-H, Shih J-S. Using fuzzy logic and particle swarm optimization to design a decision-based filter for cDNA microarray image restoration. Eng Appl Artif Intell. 2014;36:12–26.
22. Mood S, Rasshedi E, Javidi M. New functions for mass calculation in gravitational search algorithm. J Comput Sec. 2016. 2(3).
23. Modieginyane KM, Letswamotse BB, Malekian R, Abu-Mahfouz AM. Software defined wireless sensor networks application opportunities for efficient network management: A survey. Computers & Electrical Engineering. 2018 Feb 1;66:274-87.
24. Nie F, Zeng Z, Tsang IW, Xu D, Zhang C. Spectral embedded clustering: a framework for in-sample and out-of-sample spectral clustering. IEEE Trans Neural Netw. 2011;22(11):1796–808.
25. Heinzelman WR, Chandrakasan A, Balakrishnan H. Energy-efficient communication protocol for wireless microsensor networks. In System sciences, 2000. Proceedings of the 33rd annual Hawaii international conference on. 2000. IEEE.
26. Muruganathan SD, Ma DCF, Bhasin RI, Fapojuwo AO. A centralized energy-efficient routing protocol for wireless sensor networks. IEEE Commun Mag. 2005;43(3):S8–13.
27. Pradhan N, Sharma K, Singh VK. A survey on hierarchical clustering algorithm for wireless sensor networks. Energy. 2016;134(4):30–5.
28. Curry RM, Smith JC. A survey of optimization algorithms for wireless sensor network lifetime maximization. Comput Ind Eng. 2016;101:145–66.
29. Latiff NA, Tsimenidis CC, Sharif BS. Energy-aware clustering for wireless sensor networks using particle swarm optimization. Personal, Indoor and Mobile Radio Communications, 2007. PIMRC 2007. IEEE 18th International Symposium on. 2007. IEEE.
30. Mirhosseini M, Barani F, Nezamabadi-pour H. QQIGSA: a quadrivalent quantum-inspired GSA and its application in optimal adaptive design of wireless sensor networks. J Netw Comput Appl. 2017;78:231–41.
31. Bäck T, Hoffmeister F. Extended selection mechanisms in genetic algorithms. 1991.
32. Blickle T, Thiele L. A comparison of selection schemes used in genetic algorithms. 1995, TIK-report.
33. Whitley LD. The genitor algorithm and selection pressure: why rank-based allocation of reproductive trials is best. in ICGA. 1989. Fairfax, VA.
34. Yao X, Liu Y, Lin G. Evolutionary programming made faster. IEEE Trans Evol Comput. 1999;3(2):82–102.
35. Sastry K, Goldberg D, Kendall G. Genetic algorithms, in Search methodologies. 2005, Springer. 97–125.
36. Friedman M. The use of ranks to avoid the assumption of normality implicit in the analysis of variance. J Am Stat Assoc. 1937;32(200): 675–701.
37. García S, Fernández A, Luengo J, Herrera F. Advanced nonparametric tests for multiple comparisons in the design of experiments in computational intelligence and data mining: experimental analysis of power. Inf Sci. 2010;180(10):2044–64.
38. Abdi, H., Binomial distribution: binomial and sign tests. Encyclopedia of measurement and statistics, 2007. 1.
39. Zhang J, Sanderson AC. JADE: adaptive differential evolution with optional external archive. IEEE Trans Evol Comput. 2009;13(5): 945–58.
40. Shi Y, Eberhart R. A modified particle swarm optimizer. In Evolutionary Computation Proceedings, 1998. IEEE World Congress on Computational Intelligence., The 1998 IEEE International Conference on. 1998. IEEE.
41. Tsai H-C, Tyan YY, Wu YW, Lin YH. Gravitational particle swarm. Appl Math Comput. 2013;219(17):9106–17.
42. Sarafrazi S, Nezamabadi-Pour H, Saryazdi S. Disruption: a new operator in gravitational search algorithm. Scientia Iranica. 2011;18(3):539–48.
43. Li X, Engelbrecht A, Epitropakis MG. Benchmark functions for CEC'2013 special session and competition on niching methods for multimodal function optimization. RMIT University, Evolutionary Computation and Machine Learning Group, Australia, Tech. Rep, 2013.
44. Liang J et al. Problem definitions and evaluation criteria for the CEC 2013 special session on real-parameter optimization. Computational Intelligence Laboratory, Zhengzhou University, Zhengzhou, China and Nanyang Technological University, Singapore, Technical Report, 2013. 201212: 3–18.
45. Rodríguez-Fdez I et al. STAC: a web platform for the comparison of algorithms using statistical tests. In Fuzzy Systems (FUZZ-IEEE), 2015 IEEE International Conference on. 2015. IEEE.
46. An J, Kang Q, Wang L, Wu Q. Mussels wandering optimization: an ecologically inspired algorithm for global optimization. Cogn Comput. 2013;5(2):188–99.
47. Eberhart R, Kennedy J. A new optimizer using particle swarm theory. In Micro Machine and Human Science, 1995. MHS '95., Proceedings of the Sixth International Symposium on. 1995. IEEE.
48. He S, Wu QH, Saunders J. Group search optimizer: an optimization algorithm inspired by animal searching behavior. IEEE Trans Evol Comput. 2009;13(5):973–90.
49. Kumar S. Energy efficient clustering algorithm for WSN. In Signal Processing and Integrated Networks (SPIN), 2015 2nd International Conference on. 2015. IEEE.
50. Mekonnen MT, Rao KN. Cluster optimization based on metaheuristic algorithms in wireless sensor networks. Wirel Pers Commun. 2017;97(2):2633–47.
51. RejinaParvin J, Vasanthanayaki C. Particle swarm optimization-based clustering by preventing residual nodes in wireless sensor networks. IEEE Sensors J. 2015;15(8):4264–74.
52. Kennedy J. Particle swarm optimization. Encyclopedia of machine learning. 2011, Springer. 760–766.