Engineering Applications of Artificial Intelligence 94 (2020) 103731

Contents lists available at ScienceDirect

Engineering Applications of Artificial Intelligence

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

Lévy flight distribution: A new metaheuristic algorithm for solving engineering optimization problems

Essam H. Houssein [a], Mohammed R. Saad [b], Fatma A. Hashim [c], Hassan Shaban [a], M. Hassaballah [d,*]

[a] *Faculty of Computers and Information, Minia University, Egypt*
[b] *Faculty of Computers and Information, Luxor University, Luxor, Egypt*
[c] *Department of Biomedical Engineering, Faculty of Engineering, Helwan University, Helwan, Egypt*
[d] *Department of Computer Science, Faculty of Computers and Information, South Valley University, Qena, Egypt*

ARTICLE INFO

Keywords:
Engineering optimization problems
Evolutionary computation
Global optimization
Lévy flight distribution
Metaheuristic
Wireless sensor networks

ABSTRACT

In this paper, we propose a new metaheuristic algorithm based on Lévy flight called Lévy flight distribution (LFD) for solving real optimization problems. The LFD algorithm is inspired from the Lévy flight random walk for exploring unknown large search spaces (e.g., wireless sensor networks (WSNs). To assess the performance of the LFD algorithm, various optimization test bed problems are considered, namely the congress on evolutionary computation (CEC) 2017 suite and three engineering optimization problems: tension/compression spring, the welded beam, and pressure vessel. The statistical simulation results revealed that the LFD algorithm provides better results with superior performance in most tests compared to several well-known metaheuristic algorithms such as simulated annealing (SA), differential evolution (DE), particle swarm optimization (PSO), elephant herding optimization (EHO), the genetic algorithm (GA), moth-flame optimization algorithm (MFO), whale optimization algorithm (WOA), grasshopper optimization algorithm (GOA), and Harris Hawks Optimization (HHO) algorithm. Furthermore, the performance of the LFD algorithm is tested on other different optimization problems of unknown large search spaces such as the area coverage problem in WSNs. The LFD algorithm shows high performance in providing a good deployment schema than energy-efficient connected dominating set (EECDS), A3, and CDS-Rule K topology construction algorithms for solving the area coverage problem in WSNs. Eventually, the LFD algorithm performs successfully achieving a high coverage rate up to 43.16 %, while the A3, EECDS, and CDS-Rule K algorithms achieve low coverage rates up to 40 % based on network sizes used in the simulation experiments. Also, the LFD algorithm succeeded in providing a better deployment schema than A3, EECDS, and CDS-Rule K algorithms and enhancing the detection capability of WSNs by minimizing the overlap between sensor nodes and maximizing the coverage rate. The source code is currently available for public from: https://www.mathworks.com/matlabcentral/fileexchange/76103-lfd.

1. Introduction

An inseparable part of science and engineering is how to solve the optimization problems especially with the emergence of new technologies such as whose based on artificial intelligence (Ojha et al., 2017). Thus, numerical optimization problems have become increasingly complex and require highly efficient methods to solve. For example, design cost problems in engineering and accuracy problems often demand methods to find optimum from a large number of available solutions, without wasting efforts in searching sub-optimal regions (Hayyolalam and Kazem, 2020). Due to complex nature and highly non-convex landscapes, the search-space related to these problems poses several challenges to optimization methods. These include exceptionally complex modalities of the search environment; proportional to the problem size, as well as, growing problem dimensionality (Hussain et al., 2019). The conventional deterministic methods, based on simple calculus rules perform exhaustive search, cannot produce as efficient solutions as heuristic methods within limited computational resources (Sulaiman et al., 2020).

Optimization is identified as the process of selecting the best possible solutions for a given scientific/engineering problem globally or almost globally. To solve an optimization problem, four steps must be performed. First, identifying the parameters of a problem and based on these parameters the optimization problem can be categorized as discrete or continuous. Second, one or more constraints must be recognized for that controlling the parameters (Shadravan et al., 2019). The constraints separate optimization problems into two categorizes: constrained optimization and unconstrained optimization. Third, goals of the problem must be scrutinized and considered. The optimization

[*] Corresponding author.
E-mail address: m.hassaballah@svu.edu.eg (M. Hassaballah).

https://doi.org/10.1016/j.engappai.2020.103731
Received 8 January 2020; Received in revised form 11 April 2020; Accepted 25 May 2020
Available online 16 June 2020

problems are labelled into single and multi objective tasks (Cheraghalipour et al., 2018; Marler and Arora, 2004). Finally, a compatible optimizer must be identified and used to resolve the optimization problem depending on the constraints, parameters types, and goals number. Often, most real world optimization problems are severely nonlinear, not continuous so it includes complex constraints and many design variables.

Several metaheuristic algorithms have been developed over the last ten years to solve difficult and complex non-linear problems of optimization. Generally speaking, traditional optimization schemes use for solving the optimizing problems are mostly deterministic that suffer from major issues such as the local optima, unbalanced exploration/exploitation, and the need to derivate search space (Simpson et al., 1994). Due to these drawbacks, a growing interest has been observed in stochastic optimization techniques (Boussaïd et al., 2013). Metaheuristic algorithms contribute significantly to the creation of multi-agent systems and have properties of self-organization, durability, coordination, simplicity, and distribution (Yang et al., 2018). Examples of recent metaheuristic algorithms are Henry gas solubility optimization (Hashim et al., 2019), elephant herding optimization (EHO) (Wang et al., 2015), whale optimization algorithm (WOA) (Mirjalili and Lewis, 2016), grasshopper optimization algorithm (GOA) (Saremi et al., 2017), Harris hawk optimization algorithm (Heidari et al., 2019), and moth-flame optimization algorithm (MFO) (Mirjalili, 2015). All these algorithms exhibit on the average an equivalent performance if they were applied to solve all potential optimization problems (Heidari et al., 2019).

On the other hand, Lévy flights (LF) have been considered in a number of optimizers to improve their performance. As an example, Lévy flights were employed in the whale optimization algorithm (Ling et al., 2017). Also, LF has been combined with a number of algorithms, such as the Grey wolf optimizer (Heidari and Pahlavani, 2017), the ant lion optimizer for optimization problems (Dinkar and Deep, 2018), and the cultural algorithm for real parameter optimization (Ali et al., 2018) to determine new search agent position. For feature selection problems, a Lévy flight based antlion optimization (ALO) algorithm is proposed in Emary and Zawbaa (2019). Subsequently, the choice of the LF approach could more be argued and positioned regarding literature and practise. Because of its high ability in enhancing the performance of the optimization algorithms in exploration and exploitation phases as well as escaping from local optima. LF considers a free random walk or a family of scale-free walks by random steps and directions based on the Lévy distribution. LF can be used in constructing new optimization algorithms based on its various inspirations and mathematical models. Thus, it has been applied to many optimization algorithms to enhance their performance and hunting behaviour, rendering them more pragmatic. Additionally, LF can increase the exploration ability of these algorithms (Emary et al., 2019).

It is well known that existing metaheuristic algorithms cannot solve all types of optimization problems (Wolpert, 1997). In other words, there is no one algorithm perfectly qualified for solving all kinds of optimization problems in the same time, which logically opens the door for promising developments in the design and proposal of new optimization algorithms, continuously. Hence, there is always a need for new or enhanced optimization algorithms to better resolve existing and future problems that are too complicated to solve with current methods (Gupta and Deep, 2019). To this end, a new metaheuristic algorithm called Lévy flight distribution (LFD) algorithm is proposed using the Lévy flight concept within the simulation environment of wireless sensor network environment. The effectiveness of the proposed LFD algorithm is demonstrated using 22 mathematical test functions. Also, it is used for solving several optimization problems including the congress on evolutionary computation (CEC) 2017 suite and three challenging engineering design problems. Experimental results revealed that LFD is superior to other well-known metaheuristic algorithms in the literature. Further, the area coverage problem in wireless sensor networks (WSNs) as a real application is solved, where the LFD provides a better deployment schema than common topology construction algorithms, namely A3 (Wightman and Labrador, 2008), CDS-Rule *K* (Wu et al., 2006), and energy efficient connected dominating set (EECDS) (Yuanyuan et al., 2006). The main contributions of this paper are as follows.

1. A new metaheuristic algorithm called Lévy flight distribution (LFD) algorithm is proposed for solving a number of different engineering optimization problems.
2. The performance of the LFD algorithm is analysed considering several common metaheuristic algorithms with well-known CEC 2017, three real-world engineering problems, and the deployment problem in WSNs.
3. The LFD algorithm obtained a better deployment schema for the WSN with minimize the overlap between sensor nodes than the common the topology construction algorithms, especially the A3, EECDS, and CDS-Rule K algorithms.
4. Experimental results revealed the merits of the proposed LFD algorithm compared with the other competitor metaheuristic algorithms.

The rest of this paper is organized as follows. In Section 2, a review of previous studies related to metaheuristic optimization algorithms is discussed. Section 3 describes details of the proposed LFD algorithm including the mathematical model, computational procedure, and pseudo code. Section 4 provides the experimental results of the proposed LFD algorithm in solving different benchmark and real world applications. Finally, conclusions are given in Section 5.

2. Literature review

Recently, several metaheuristic algorithms have been introduced to solve a large array of real life problems (Houssein et al., 2020b,a). The majority of algorithms are nature inspired and imitate some feature principles of biological systems, physical systems, ethology or swarm intelligence (Çelik, 2020; LaTorre et al., 2015). Surprisingly, some of such algorithms including genetic algorithm (GA) (Bonabeau et al., 1999), and particle swarm optimization (PSO) (Eberhart and Kennedy, 1995b) are quite known among not only computer experts but also experts from other research fields. In fact, the wide spread of the metaheuristic algorithms especially in engineering optimization problems is back to some reasons: flexibility, gradient-free mechanism, and local optima avoidance relying on simple concepts (Ab Wahab et al., 2015). Most nature-inspired metaheuristics algorithms are almost simple because they are usually inspired by uncomplicated or straightforward concepts.

Metaheuristic algorithms can roughly classified into four main categories: evolutionary based, swarm based, physics or chemistry based and human behaviour based algorithms (Kaur et al., 2020; Zhang et al., 2018). Evolutionary based algorithms are just an imitation to evolutionary processes in the nature (Back, 1996). In this type of algorithms, the global optima are reached through generating a new offspring that inherits characteristics and properties of parents through picking out agents randomly from the current population as parents and involves them to produce offspring for next generation (Simon, 2013). Evolutionary Strategies (Amoretti, 2014), Genetic Algorithm (Holland, 1992; Tang et al., 1996), and genetic programming (GP) (Langdon and Poli, 2013) are common algorithms of evolutionary based algorithms. Though they solved some optimization problems, such as infinite monkey theorem, Richard Dawkin's weasel, and travelling salesman problem, the computational complexity is the main drawback of these algorithms (Amoretti, 2014).

Swarm based algorithms emulate the social and intelligent behaviour of a set of creatures (e.g., birds, insects, fish). For instance, the popular particle swarm optimization (Eberhart and Kennedy, 1995b,a) was inspired from the movement of birds and a recent moth-flame optimization algorithm (Mirjalili, 2015) was inspired from the navigation manner of moths. Swarm based algorithms have the properties of self-organization, durability, coordination, simplicity, and distribution applies to solve optimization problems. Also, they have the characteristics of sharing information among multiple agents, self-organized, co-evolution, learning during iterations to perform efficient search operations (Yang et al., 2018). Besides, different agents may be parallelized so that large scale optimization becomes more practical from the implementation opinion. Artificial bee colony (Karaboga et al., 2014), ant colony optimization (Dorigo et al., 2006), whale optimization algorithm (Mirjalili and Lewis, 2016), grasshopper optimization algorithm (Saremi et al., 2017), dragonfly algorithm (Mirjalili, 2016), Harris hawks optimization (Heidari et al., 2019), artificial fish swarm algorithm (Neshat et al., 2014), elephant herding optimization (Wang et al., 2015), and glowworm swarm optimization (Krishnanand and Ghose, 2009) are some examples of swarm based algorithms.

Physics or chemistry based metaheuristic algorithms are designed differently, where inspirations come from observed phenomena in physics or chemistry. Normally, these algorithms simulate some physical or chemical laws, such as electrical charges, river systems, chemical reactions, gas pressure, gravity, etc. For instance, the gravitational search algorithm introduced in Rashedi et al. (2009) simulates the idea of Newton's theory of gravitation while the chemical reaction algorithm (Truong et al., 2013) mimics chemical reactions. The equilibrium optimization algorithm (Faramarzi et al., 2020) mimics the estimation process of equilibrium states using control volume mass balance models. The magnetic charged system search (Kaveh et al., 2013), ions motion algorithm (Javidy et al., 2015), atom search optimization (Zhao et al., 2019), henry gas solubility optimization (Hashim et al., 2019) are belong to the family of physics/chemistry based metaheuristic algorithms.

The Human behaviour based algorithms mimic behaviour of human beings including social and emotional behaviour, competition, teaching, and learning (Sulaiman et al., 2020). The social-emotional optimization algorithms (Guo et al., 2017; Ramezani and Lotfi, 2013) mimic the human social emotional behaviour. The inspiration of teaching-learning based optimization algorithm (Ouyang et al., 2015) came from the behaviour followed in teaching and learning students in classrooms. The volleyball premier league algorithm (Moghdani and Salimifard, 2018) mimics the competition, interaction between teams of volleyball and the coaching process in the match. A recent algorithm from this metaheuristic algorithm family is the fibonacci indicator algorithm (Etminaniesfahani et al., 2018), which mimics technical traders in stock markets based on predicting possible local maximum and minimum prices, and periods of significant movements in the prices of stocks. These algorithms achieve competitive and promising performance in several applications. Unfortunately, they suffer from slow convergence rates in most cases.

Besides these four categories, there are a large number of hybrid metaheuristics algorithms, which also are able to provide very promising and competitive results with accurate solutions in many challenging optimization problems. In this regard, to improve the performance of the particle swarm optimization algorithm in the application of analysing patient attendance to hospitals, a combination of clustering algorithm and PSO is introduced in Liu et al. (2018). In Okulewicz and Mańdziuk (2019), both differential evolution and PSO algorithms are utilized for solving dynamic vehicle routing problems. In Xu et al. (2019), two strategies were integrated into moth-flame optimization algorithm (MFO) (Mirjalili, 2015) to achieve balance between the exploration and exploitation. The interactive search algorithm (Mortazavi et al., 2018) combined features of integrated particle swarm optimization and teaching and learning based optimization algorithms. In spite of the vast amount of works and studies on this field, solving optimization problems is not quite solved and it remains a very challenging task for a long time.

On the other hand, a great number of complex and challenging optimization problems have arisen in different fields due to emergence of new technologies. For example, recent advances in embedded systems and radio commutations have enabled the proliferation of wireless sensor networks (WSNs) (Mann and Singh, 2017). One of the most common problems in WSNs is area coverage, which results from the random deployment of various topology control algorithms. This problem will negatively affect the efficiency and detection capability of a network. The negative effect is represented in the redundancy area with a low coverage ratio to decrease the damage to WSNs (Yiyue et al., 2012). Therefore, some metaheuristic algorithms have been presented and employed to tackle the issue of area coverage in WSNs (Ahmed et al., 2019; Gupta and Jha, 2018). In Liao et al. (2011), a deployment schema for WSN based on glowworm swarm optimization was introduced through applying the concept of luciferin, which is a luminant substance emitting from individual glowworms, and the glowworms move based on the intensity of this substance. Each glowworm is simulated with each sensor while maintaining a certain distance from its neighbour to avoid overlap and to achieve the highest possible area coverage. This deployment schema showed that it can provide good coverage with limited movement. In Ly et al. (2015), an improved genetic algorithm was presented to ensure the maximum area coverage in WSNs of certain number of sensors with different sensing ranges. In Binh et al. (2018), another deployment schema for WSN was provided using improved cuckoo search and chaotic flower pollination optimization algorithms for maximizing the area coverage of the network and minimizing the overlap between sensor nodes, which causes high damage to the network.

3. Lévy flight distribution (LFD)

This section introduces source of inspiration, mathematical model, and pseudo-code of the proposed optimization algorithm.

3.1. Inspiration

The proposed algorithm is based mainly on the wireless sensor networks environment jointing with LF motions. LF can be considered a random walk as shown in Fig. 1. In uncertain environments, the LF showed that it can increase the effectiveness of resource searches. In fact, LFs can be inspired by many natural-inspired or physical-inspired phenomena in the environment. Natural species such as spider monkeys, fruit flies, and humans especially in the Ju/'hoansi gatherers for hunters, can trail the paths of LF styles. Additionally, foraging patterns of albatrosses have been observed as an inspiration phase for the LFs. The diffusion of fluorescent molecules can be considered a physically-inspired phenomenon for the inspiration of LFs, and noise and cooling behaviours show the characteristics of LFs under the right conditions. Furthermore in exploring unknown large search spaces, LFs are more efficient compared to Brownian random walks (Magdziarz and Szczotka, 2016; Yang et al., 2013); this efficiency is the major motivation that inspired us to consider LFs in the proposed optimization algorithm.

3.2. Mathematical model

The simulation environment for modelling the following mathematical model is the wireless sensor network environment in which the algorithm starts its mechanism with calculating the Euclidean distance ED between every two adjacent sensor nodes and then the algorithm determines whether the sensor node will be in its original position based on the calculated distance ED or moving it in another position. Another position will be performed using LFs model in which the new position will be in a place which closes to a sensor node that has a low amount of neighbours nodes or in a place which has no sensor nodes in the deployment area (search space) to decrease the opportunities for occurring the overlapping among sensor nodes.

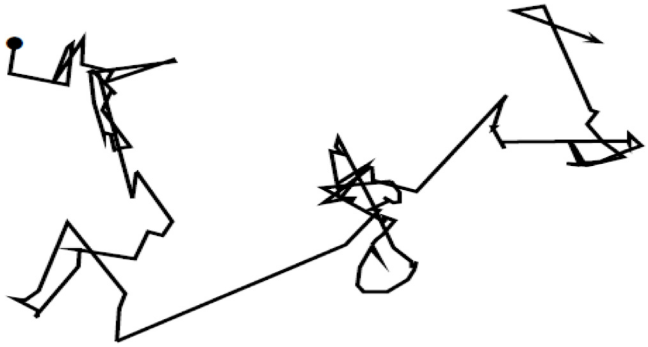


Fig. 1. Lévy flights of fifty sequential steps starting from the origin marked with a bold point.

For implementing LFs or generating random walks, two characteristics must be identified: the step length of this walk that follows the chosen Lévy distribution and the direction that makes it move towards the target position in the proposed algorithm, which can be derived from the uniform distribution. Many methods can determine these features, but the most simple and efficient one is the well-known Mantegna algorithm for a symmetric and stable Lévy distribution (Yang et al., 2013). According to Mantegna's algorithm, the step length S is defined as

$$S = \frac{U}{|V|^{1/\beta}} \tag{1}$$

where, β is the Lévy distribution index bounded as $0 < \beta \leq 2$, while U and V are such that

$$\mathrm{U} \sim \mathrm{N}\left(0, \sigma_{\mathrm{u}}^2\right), \quad \mathrm{V} \sim \mathrm{N}\left(0, \sigma_{\mathrm{v}}^2\right), \tag{2}$$

The standard deviation σ_u and σ_v are defined by

$$\sigma_u = \left\{ \frac{\Gamma(1+\beta) \times \sin(\pi\beta/2)}{\Gamma[(1+\beta)/2] \times \beta \times 2^{(\beta-1)/2}} \right\}^{1/\beta}, \quad \sigma_v = 1. \tag{3}$$

where the Gamma function Γ for an integer z is expressed as

$$\Gamma(z) = \int_0^\infty t^{z-1} e^{-t} dt. \tag{4}$$

Mathematically speaking, the proposed LFD algorithm calculates the Euclidean distance ED between positions of the first two adjacent agents, namely X_i and X_J (the neighbour of X_i) using

$$ED\left(X_i, X_J\right) = \sqrt{\left(x_J - x_i\right)^2 + \left(y_J - y_i\right)^2} \tag{5}$$

where (x_i, y_i) is the position coordinate of X_i and (x_J, y_J) is the X_J position coordinate. The distance ED is compared with a given threshold until the agents are finished after a specific number of iterations. If the resultant distance is less than the threshold (this means there is a problem in deploying the agents in the search space), then the algorithm starts its mechanism by adjusting the positions of these agents using

$$X_J(\mathrm{t}+1) = Lévy\_Flight(X_J(\mathrm{t}), X_{Leader}, LB, UB) \tag{6}$$

where t is an index for number of iterations, $Lévy\_Flight$ is a function that performs the work of Lévy flights in terms of the step length and direction. Algorithm 1 illustrates the pseudo-code of this function. UB and LB are the highest and lowest values in the 2D dimensions of the search space, respectively. X_{Leader} is the position of agent that has the lowest number of neighbours and will be used as the LF direction. Eq. (6) moves the agent X_J towards the agent's position that has the lowest number of neighbours.

$$X_J(\mathrm{t}+1) = LB + (UB - LB)\ rand(\ ) \tag{7}$$

Algorithm 1 Pseudo code of the $Lévy\_Flight$ function.

```
Input: $CurrentPosition$, $DirectionPosition$, $LB$, $UB$.
Output: $NewPosition$.
Begin
Calculate the step length $S$ based on Mantegna's algorithm using Eq. (1)
Calculate the difference factor $DF = 0.01 \times S \times [CurrentPosition - DirectionPosition]$
Execute the actual random walk or flight with $NewPosition = CurrentPosition + S \times rand(\ ) \times [size(CurrentPosition)]$
Apply the simple bounds on $NewPosition$ to ensure that it does not go outside the search space.
Return $NewPosition$
End.
```

where $rand(\ )$ is a function utilized to generate random numbers R in the uniform distribution $[0, 1]$. Eq. (7) presents more chances for discovering non-visited position solutions in the search space and increasing the exploration phase of the proposed algorithm. This equation updates the position of X_J to a new region in the search space where there are no other agents.

$$R = rand(\ ), \quad CSV = 0.5 \tag{8}$$

where CSV is the comparative scalar value with R in each update for the X_J position. To update the sensor node X_J position the algorithm at each iteration checks on the value of R in Eq. (8). If the value is less than CSV, then execute Eq. (6). If not, then execute Eq. (7) to give the algorithm more opportunities to discover the search space. Varying the algorithm's solutions will increase its exploration ability and enhance its performance. The proposed algorithm updates the position of X_i, using

$$X_i(t+1) = TP + \alpha_1 \times TF_{Neighbours} + rand(\ ) \times\ \alpha_2 \times ((TP + \alpha_3 X_{Leader})/2 - X_i(\mathrm{t})) \tag{9}$$

$$X_i^{New}(t+1) = Lévy\_Flight\left(X_i(t+1), TP, LB, UB\right) \tag{10}$$

The new position X_i is calculated by Eq. (9), while the final position for X_i is given by Eq. (10). The $Lévy\_Flight$ is a function illustrated in Algorithm 1 and TP is the position (solution) that achieved the best fitness value of the objective function, which is called the target position. The random numbers α_1, α_2 and α_3 are such that $0 < \alpha_1,\ \alpha_2,\ \alpha_3 \leq 10$. The total target fitness of neighbours $TF_{Neighbours}$ around $X_i(t)$ is given by

$$TF_{Neighbours} = \sum_{k=1}^{NN} \frac{\mathrm{D(k)} \times X_k}{NN} \tag{11}$$

where X_k is the neighbour position of $X_i(t)$, K is the neighbour's index, NN is the total number of $X_i(t)$ neighbours, and $D(K)$ is the fitness degree for each neighbour given by

$$D(k) = \frac{\partial_1 (V - Min(V))}{Max(V) - Min(V)} + \partial_2 \tag{12}$$

where,

$$V = \frac{Fitness(X_J(\mathrm{t}))}{Fitness(X_i(\mathrm{t}))}, \quad and\ \ 0 < \partial 1,\ \ \partial 2 \leq 1. \tag{13}$$

A schematic flowchart for the proposed LFD algorithm is illustrated in Fig. 2 and the pseudo code is presented in Algorithm 2. All the previously defined equations will be repeated at each iteration in the LFD algorithm. Suppose t is the total number of iterations and n is the number of agent. It clear that the time complexity of LFD is of $O(tnd)$,

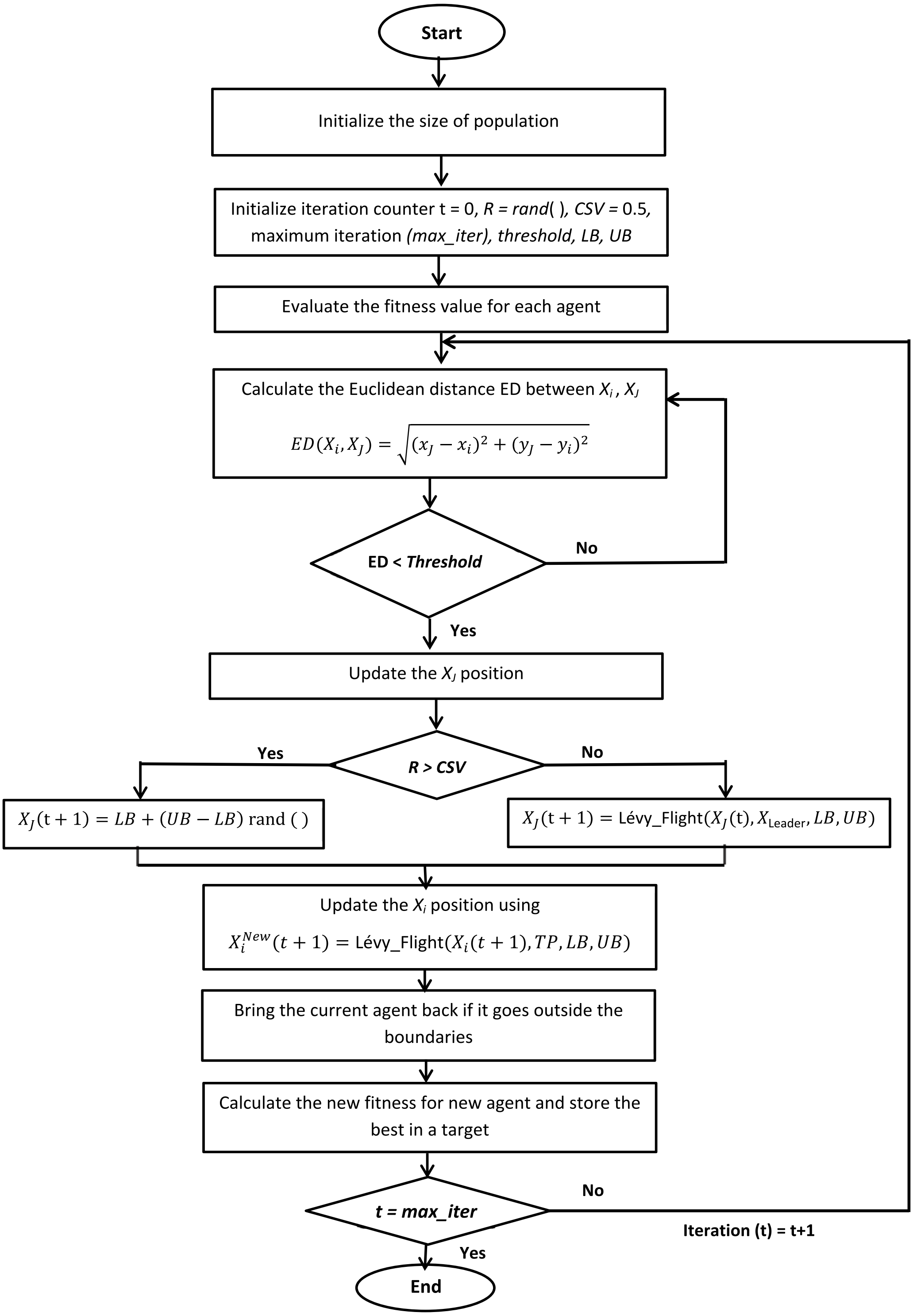

Start
Initialize the size of population
Initialize iteration counter t = 0, R = rand( ), CSV = 0.5, maximum iteration (max_iter), threshold, LB, UB
Evaluate the fitness value for each agent
Calculate the Euclidean distance ED between X_i, X_J
$ED(X_i, X_J) = \sqrt{(x_J - x_i)^2 + (y_J - y_i)^2}$
ED < Threshold
No
Yes
Update the X_J position
R > CSV
Yes
No
$X_J(\text{t}+1) = LB + (UB - LB)$ rand ( )
$X_J(\text{t}+1) =$ Lévy_Flight(X_J(t), X_{Leader}, LB, UB)
Update the X_i position using
$X_i^{New}(t+1) =$ Lévy_Flight($X_i(t+1)$, TP, LB, UB)
Bring the current agent back if it goes outside the boundaries
Calculate the new fitness for new agent and store the best in a target
t = max_iter
No
Iteration (t) = t+1
Yes
End

Fig. 2. A schematic flowchart for the proposed LFD algorithm.

where d refers to the dimension of each agent. This is the time complexity of the proposed algorithm using the BigO notation without paying attention to objective functions. Therefore, the total complexity of the solution including the objective function (obj) (represented by the value of $Fitness(X)$ defined in Eq. (12)) is $O(tnd) * O(obj)$. Advantages of the proposed algorithm are its high exploitation and exploration abilities, escaping from local optima, and balancing between exploration and exploitation. On the other side, it has a disadvantage comparing with other well-known algorithms related to time complexity. Although the aforementioned theoretical discussions proved the efficiency of the LFD algorithm in finding the global optimum, in the following sections, the performance of the LFD algorithm is explored experimentally on several challenging real engineering optimization problems.

4. Experiments and discussions

This section presents performance validation and experimentation of the proposed algorithm on 22 standard benchmark test functions. Detailed description for these benchmark test functions as well as parameters settings and performance evaluation metrics are discussed

Algorithm 2 Pseudo code of the proposed LFD algorithm.

```
Initialize the size of population $X_i, i = 1, 2, \dots, n$.
Initialize LB, UB, Threshold, and $Max\_iter$.
Calculate the fitness for each agent.
$T$ = the best solution.
Output target position and target fitness.
while $t <= Max\_iter$ do
    for each two adjacent agents (main , neighbour) do
        Calculate Euclidean distance using Eq. (5)
        if Euclidean distance < Threshold then
            Update the neighbour agent position using Eqs. (6) – (7).
            Update the main agent position using Eq. (10).
            Bring the current agent back if it goes outside the boundaries whether main or neighbour.
        end if
    end for
    Update $T$ if there is a good solution.
    $t = t + 1$
end while
Return $T$
```

below. Furthermore, the performance is analysed and compared with several well-known metaheuristics algorithms on a number of real-world optimization engineering problems. The experiments has been done using Matlab software version R2013b on a PC of 32-bit Core i5 processor with 3.20 GHz and 8 GB main memory.

4.1. Parameters settings

Table 1 lists the parameters settings used in the experiments for all the competitive algorithms. Also, the number of search agents, the maximum iterations number, and the dimension are set to be 35,1000, and 10, respectively. For statistical analysis, 30 independent runs per algorithm are conducted on each benchmark problem for all examined algorithms.

4.2. Performance evaluation metrics

To measure the performance of the proposed LFD algorithm, the following evaluation criteria are used

1. Mean: average of fitness values produced by algorithms after a number of iterations M given by

$$Mean = \frac{\sum_{i=1}^{M} (f_i)}{M} \tag{14}$$

2. Standard derivation (SD): the difference between values of the objective function resulting from executing the algorithm M times. The standard derivation is calculated by

$$Std = \sqrt{\frac{1}{M-1} \Sigma_{i=1}^{M} \left(f_i - mean\right)^2} \tag{15}$$

3. Best: the minimum fitness value obtained from executing the algorithm M times expressed by

$$Best = \min_{1 \le i \le M} f_i \tag{16}$$

4. Worst: the maximum fitness value obtained from executing the algorithm M times and it takes the form

$$Worst = \max_{1 \le i \le M} f_i \tag{17}$$

where f_i is the best fitness value obtained at run i.

Table 1
Controlling parameters values of the tested algorithms.

Algorithms	Parameters	Values
GA	Chromosomes	35
	Pc	0.85
	Pm	0.01
	Er	0.05
DE	Population Size (Colony Size)	35
	F	0.9
	Lower Bound of Scaling Factor ($Beta_M in$)	0.2
	Upper Bound of Scaling Factor ($Beta_M ax$)	0.8
	Crossover Probability (PCR)	0.5
SA	Materials	35
	Initial temperature (T_init)	1.0
	Final stopping temperature (T_min)	1e−10
	Min value of the function (F_min)	−1e+100
	Boltzmann constant (K)	1
	Cooling factor ($alpha$)	0.95
	Energy norm ($Enorm$)	1e−2
PSO	Swarm size	35
	Inertia weight	∈ [0.9, 0.4]
	C_1 (individual-best acceleration factor)	∈ [.5, 2.5]
	C_2 (global-best acceleration factor)	∈ [2.5, 0.5]
EHO	Elephants number	35
	Clans number	5
	Number of elephants in each clan	7
	The scale factor alpha	0.5
	The scale factor beta	0.1
MFO	Flames number	35
	b	6
WOA	Whales number	35
	a	∈ [0, 2]
	$a2$	∈ [−1, −2]
GOA	Grasshoppers number	35
	Intensity of attraction f	0.5
	Attractive length scale l	1.5
	$Cmin$	0.00004
	$Cmax$	1
HHO	Harris Hawks Number	35
	β	1.5
	E_0 variable	∈ [−1, 1]
LFD	Search agents	35
	$Threshold$	2
	CSV	0.5
	β	1.5
	α_1	10
	α_2	0.00005
	α_3	0.005
	∂_1	0.9
	∂_2	0.1

4.3. Experiments on CEC 2017

The proposed LFD algorithm is tested on 22 functions of a well-known CEC 2017 benchmark (Wu et al., 2017). The algorithm is iterated 1000 times for every benchmark function and it simulates 30 running times with respect to dimension equals 10. In general, these benchmark test functions are divided into four main categories: unimodal, multimodal, composite functions, and hybrid functions. In this regards, the LFD algorithm is compared with several well-known algorithms, such as the GA (Holland, 1992), DE (Das and Suganthan, 2010), SA (Van Laarhoven and Aarts, 1987), PSO (Eberhart and Kennedy, 1995a), EHO (Wang et al., 2015), MFO (Mirjalili, 2015), WOA (Mirjalili and Lewis, 2016), GOA (Saremi et al., 2017), and HHO (Heidari et al., 2019) methods.

The diversity of the used benchmark functions is as follows (1) The unimodal functions F1–F3 have only one global optimum, which is used to assess the exploitation ability of the propose algorithm against the competitive metaheuristic algorithms. It is obvious from Table 2 that the LFD algorithm competes very strongly with the compared

Table 2
Experimental optimization results for dimension 10 using reported competitor algorithms on the unimodal, multimodal, hybrid, and composite of CEC 2017 benchmark functions.

		GA	DE	SA	PSO	EHO	MFO	WOA	GOA	HHO	LFD
F1	Mean	5.74E+02	5.99E+02	7.14E+02	5.49E+02	6.00E+02	5.43E+02	5.50E+02	5.82E+02	5.48E+02	5.34E+02
	SD	1.25E+01	1.91E+01	9.56E+01	3.22E+00	1.41E+01	1.37E+01	2.79E+01	1.24E+01	1.73E+01	1.01E+01
F2	Mean	6.41E+02	6.55E+02	6.86E+02	6.38E+02	6.54E+02	6.17E+02	6.29E+02	6.56E+02	6.28E+02	6.07E+02
	SD	1.06E+01	1.13E+01	2.21E+01	4.05E+00	6.36E+00	1.33E+01	1.10E+01	1.74E+01	1.20E+01	3.39E+00
F3	Mean	4.97E+04	3.39E+03	1.23E+03	1.53E+03	2.66E+03	1.60E+03	1.20E+03	6.88E+03	1.16E+03	1.15E+03
	SD	4.86E+04	3.65E+03	6.42E+01	1.16E+02	1.66E+03	3.88E+02	7.11E+01	9.20E+03	7.15E+01	4.39E+01
F4	Mean	1.34E+09	8.21E+08	5.45E+06	4.82E+07	3.28E+08	6.76E+06	2.22E+06	1.54E+08	2.58E+06	2.11E+06
	SD	5.48E+08	4.25E+08	3.40E+06	1.17E+07	2.12E+08	3.78E+06	2.32E+06	1.31E+08	2.21E+06	1.10E+06
F5	Mean	1.92E+08	2.55E+06	9.48E+04	4.68E+05	7.85E+06	1.68E+04	1.70E+04	1.53E+07	1.32E+04	1.15E+04
	SD	1.09E+08	2.91E+06	6.56E+03	4.20E+05	1.57E+07	1.56E+04	7.94E+03	3.98E+07	1.13E+04	6.13E+03
F6	Mean	2.00E+06	1.11E+06	2.48E+03	1.60E+03	8.28E+04	7.57E+03	2.82E+03	1.09E+04	1.53E+03	5.04E+03
	SD	2.27E+06	3.02E+06	3.38E+03	2.49E+01	3.61E+05	8.69E+03	2.11E+03	1.05E+04	3.64E+01	4.16E+03
F7	Mean	4.38E+07	3.93E+03	5.87E+03	8.64E+03	6.67E+04	1.60E+04	4.74E+03	6.57E+04	2.94E+03	1.62E+03
	SD	4.56E+07	2.23E+03	3.32E+03	4.60E+03	1.74E+05	1.40E+04	2.97E+03	7.19E+04	1.03E+03	1.00E+03
F8	Mean	2.62E+03	2.29E+03	2.23E+03	1.73E+03	2.16E+03	1.80E+03	1.89E+03	2.06E+03	1.79E+03	1.20E+03
	SD	1.67E+02	1.98E+02	1.82E+02	4.65E+01	9.75E+01	1.41E+02	2.69E+02	1.89E+02	7.38E+01	1.46E+01
F9	Mean	2.36E+03	1.94E+03	2.07E+03	1.88E+03	1.86E+03	1.78E+03	1.79E+03	2.05E+03	1.75E+03	1.70E+03
	SD	1.42E+02	1.04E+02	1.63E+02	3.25E+01	4.00E+01	1.24E+01	2.10E+01	1.29E+02	8.46E+00	8.09E+00
F10	Mean	5.43E+08	7.64E+07	1.68E+04	1.81E+05	6.83E+07	1.95E+04	1.61E+04	3.91E+06	2.25E+04	1.56E+04
	SD	4.53E+08	1.22E+08	9.97E+03	1.00E+05	1.36E+08	1.54E+04	5.73E+03	6.44E+06	1.09E+04	1.01E+04
F11	Mean	7.29E+07	1.03E+06	6.56E+03	7.80E+03	2.28E+06	6.05E+03	5.36E+04	4.52E+05	2.19E+03	2.01E+03
	SD	1.35E+08	2.62E+06	4.75E+03	3.25E+03	6.59E+06	3.44E+03	6.86E+04	1.35E+06	9.57E+01	4.01E+01
F12	Mean	2.33E+03	2.90E+03	2.49E+03	2.12E+03	2.26E+03	2.07E+03	2.19E+03	2.31E+03	2.12E+03	2.01E+03
	SD	1.36E+02	3.01E+01	1.78E+02	1.96E+01	4.34E+01	2.97E+01	4.07E+01	8.14E+01	9.36E+01	1.06E+01
F13	Mean	2.29E+03	2.73E+03	2.44E+03	2.35E+03	2.35E+03	2.31E+03	2.31E+03	2.36E+03	2.34E+03	2.29E+03
	SD	3.91E+01	6.50E+01	1.14E+02	2.15E+00	4.74E+01	7.95E+01	8.64E+01	4.54E+01	1.77E+01	5.85E+01
F14	Mean	2.83E+03	2.07E+03	4.10E+03	2.69E+03	2.91E+03	2.55E+03	2.32E+03	2.96E+03	2.31E+03	2.03E+03
	SD	2.35E+02	3.01E+01	6.65E+02	8.88E+01	2.03E+02	2.66E+02	6.38E+00	4.87E+02	3.31E+00	3.01E+00
F15	Mean	2.76E+03	2.32E+03	3.25E+03	2.64E+03	2.76E+03	2.64E+03	2.63E+03	2.68E+03	2.68E+03	1.63E+03
	SD	2.86E+01	4.76E+01	3.33E+02	1.63E+00	2.62E+01	1.98E+01	6.72E+00	1.56E+01	2.24E+01	1.10E+00
F16	Mean	2.81E+03	3.11E+03	2.85E+03	2.78E+03	2.79E+03	2.79E+03	2.73E+03	2.80E+03	2.88E+03	2.11E+03
	SD	1.49E+02	6.22E+02	1.96E+02	6.12E+00	5.96E+01	1.96E+00	1.56E+02	3.48E+01	7.83E+01	1.56E+00
F17	Mean	3.48E+03	2.67E+03	2.90E+03	3.06E+03	3.37E+03	3.00E+03	2.86E+03	3.24E+03	2.90E+03	1.95E+03
	SD	2.93E+02	1.56E+01	8.38E+01	2.45E+01	2.16E+02	4.52E+01	1.95E+02	2.00E+02	1.03E+00	1.01E+00
F18	Mean	4.19E+03	3.34E+03	4.31E+03	3.21E+03	3.82E+03	3.50E+03	3.47E+03	3.56E+03	3.70E+03	3.07E+03
	SD	2.94E+02	5.19E+02	9.97E+02	6.96E+01	2.14E+02	6.06E+02	8.48E+02	4.27E+02	8.09E+02	5.48E+01
F19	Mean	3.27E+03	3.10E+03	3.43E+03	3.11E+03	3.25E+03	3.10E+03	3.12E+03	3.14E+03	3.13E+03	3.10E+03
	SD	5.85E+01	6.31E+00	4.87E+02	2.81E+00	4.36E+01	6.28E+00	1.99E+01	3.80E+01	1.22E+01	6.28E+00
F20	Mean	3.63E+03	4.44E+03	3.29E+03	3.35E+03	3.70E+03	3.30E+03	3.34E+03	3.49E+03	3.21E+03	3.94E+03
	SD	1.42E+02	2.22E+02	7.61E+01	1.05E+02	1.22E+02	1.08E+02	1.05E+02	1.48E+02	6.83E+01	1.95E+02
F21	Mean	4.14E+03	3.30E+03	3.50E+03	3.23E+03	3.56E+03	3.27E+03	3.33E+03	3.40E+03	3.38E+03	2.33E+03
	SD	2.39E+02	6.24E+01	1.58E+02	1.75E+01	1.31E+02	9.59E+01	3.81E+01	1.36E+02	1.20E+02	1.21E+01
F22	Mean	1.09E+08	1.25E+06	1.08E+07	1.44E+06	2.50E+07	7.99E+05	1.52E+05	2.62E+06	4.24E+05	1.02E+05
	SD	5.94E+07	1.19E+06	1.17E+07	1.33E+06	2.63E+07	7.51E+04	6.89E+04	2.60E+06	3.42E+05	5.20E+04

metaheuristic algorithms. The LFD algorithm achieves the best results for functions F1, F2, and F3 in all tested dimensions and achieved good exploitation. (2) The multimodal functions F4–F10 differ from unimodal functions in the total number of local optima they have. The number of multimodal functions increases exponentially with the size of the problem (number of design variables). Consequently, this type of benchmark function is very useful for evaluating the exploration ability of the compared metaheuristic algorithm. The reported results in Table 2 for the functions F4–F10 reveals that the LFD algorithm has near perfect exploration ability. The LFD algorithm is always the most effective or at least the second best in the plurality of benchmark problems. This success is a result of integrated exploration techniques that are available in the LFD algorithm, which attracts it into the direction of the desirable global optimum. The composite mathematical and hybrid functions F11–F22, one of the most challenging issues, are used to measure both the exploration and exploitation ability and escape from local optimal traps of the proposed LFD algorithm and competitive algorithms. Also, the results listed in Table 2 confirms that the LFD algorithm is an effective algorithm because it provides good results and performance than other competitive algorithms in terms of the mean and standard derivation, especially for 10 dimensions.

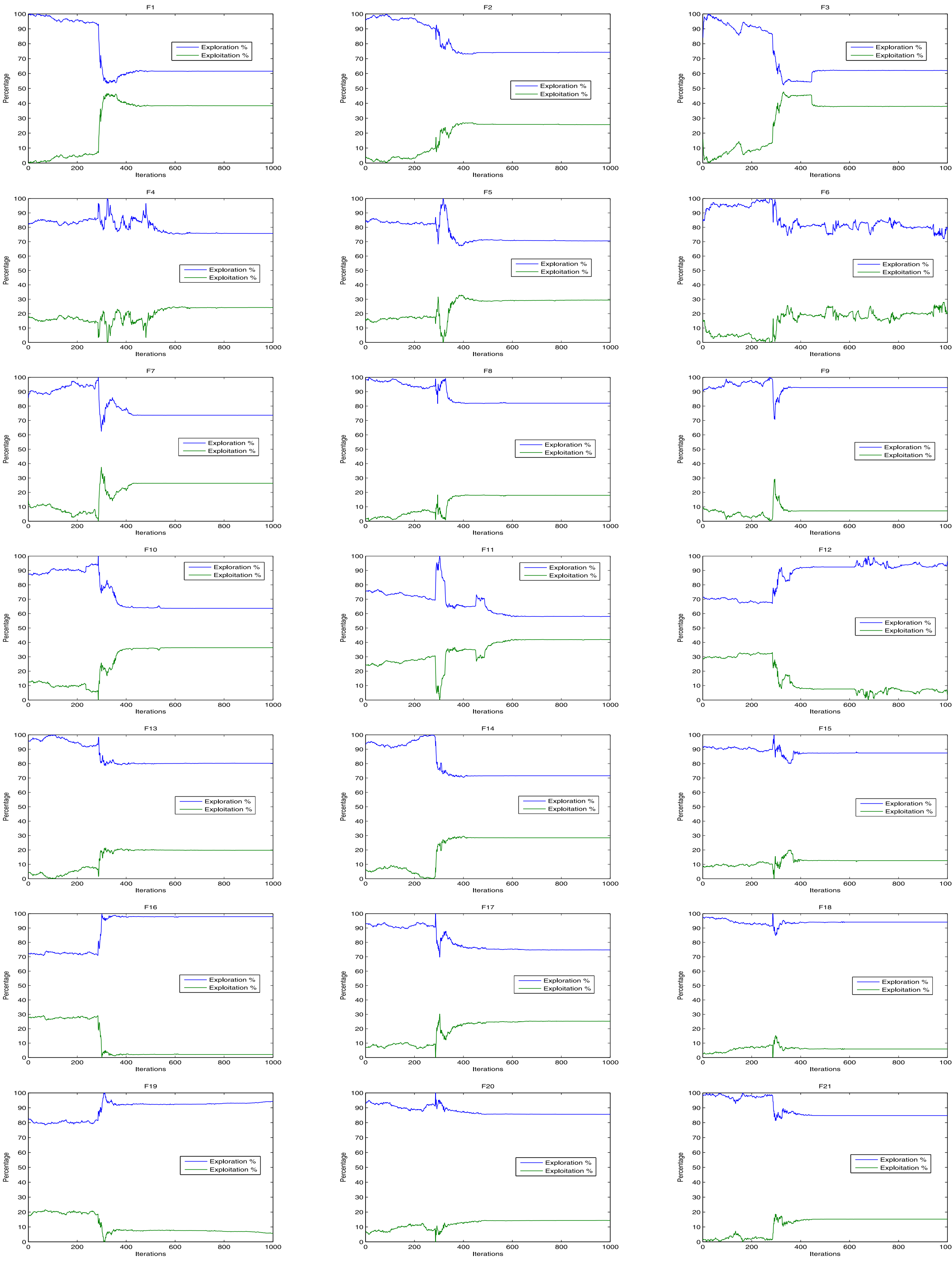


Fig. 3. Exploration and exploitation phases for the LDF algorithm on samples of benchmark functions.

4.3.1. *Exploration and exploitation phases analysis*

For more clarification, Fig. 3 depicts the average of exploration and exploitation phases of the proposed algorithm based on Eq. (18). It clear from the curves shown in Fig. 3 that the LFD algorithm achieves high exploration rates for finding the global optima. Furthermore, the LFD algorithm has a good escaping ability from local optima and it balances very well between exploration and exploitation phases.

$$\frac{1}{Div_j} = \frac{1}{N}\sum_{i=1}^{N} \text{Median}\left(x^j\right) - x_i^j; \quad Div^t = \frac{1}{N}\sum_{j=1}^{D} Div_j \tag{18}$$

where, x_i^j is jth dimension of ith population individual, Div_i refers to the mean diversity of dimension j and Div^t is computed for all D dimensions with iterations t. By computing the population diversity for all iterations $iter$, it will be able to determine easily how much percentage

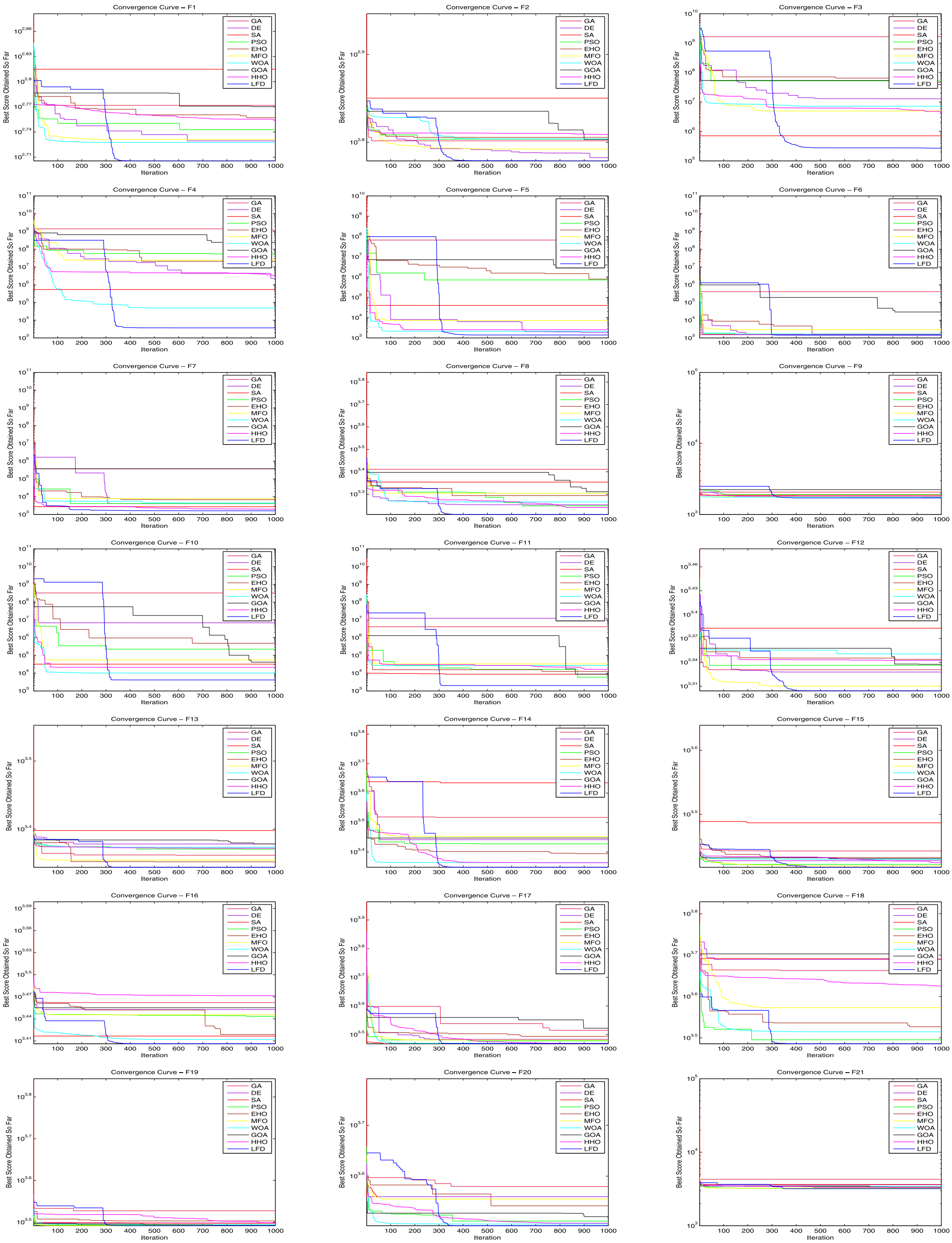


Fig. 4. Convergence curve of the algorithm on CEC 2017 test suite with 10 dimensions.

the search process is explorative and how much it is exploitative using

$$\text{Exploration}\ \% = \frac{Div^{t}}{Div_{\max}} \times 100$$
$$\text{Exploitation}\% = \frac{|Div^{t} - Div_{\max}|}{Div_{\max}} \times 100 \qquad (19)$$

where, Div_{max} is the maximum diversity for the entire iterations.

In this context, the results listed in Table 2 prove that the proposed LFD algorithm is one the best methods for the four benchmark problems and that it is very competitive in the other cases. This evidence indicates that the LFD algorithm demonstrates good ability to balance the exploration and exploitation phases. Such a dragging ability from the adaptive delineation was applied to update the X vector. Few cycles are dedicated to exploration ($|X| \geq 1$), whilst the remainder are dedicated to exploitation ($|X| < 1$).

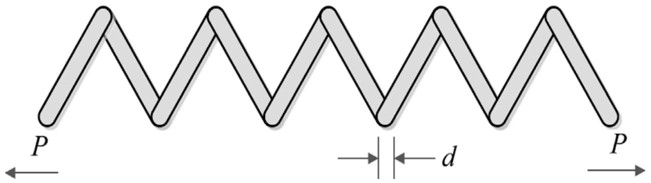


Fig. 5. A schematic illustration for the tension/compression spring problem.

4.3.2. Convergence behaviour analysis

The convergence curve is a relation between values of fitness functions and the number of iterations for all competitive algorithms. The convergence curve plots the best fitness value [f] at iteration [t], until the algorithm iterations are met. Additionally, measuring the ability of the optimization algorithm to quickly reach the solution is one of the most necessary graphical examinations for the performance evaluation of the competitive algorithms. This examination help us to analyse the convergence rates of optimization algorithms. The convergence curves of the LFD algorithm and other comparative algorithms are illustrated in Fig. 4 for 1000 iterations and 30 runs. It is obvious from Fig. 4 that the LFD algorithm clarifies three various convergence behaviours in the task of optimizing the benchmark problems. First, the convergence of the LFD algorithm accelerates its behaviour to reach the solution as the iterations increase. This effect is considered a result of the adaptive mechanisms that exist in the LFD algorithm. These adaptive mechanisms help the algorithm to explore the search space looking for the promising regions during the initial steps of the algorithm iteration and more quickly converge to the optimum after completing nearly half of the iterations. This behaviour is clear in F1, F3, F4, F5, and F9. The convergence to the optimum in the final iterations, as seen in F8 and F13, is the second behaviour of the LDF algorithm, as derived from Fig. 4.

The second behaviour probably appears as a result of a failure of the LFD algorithm to obtain a good solution for exploitation during the initial steps of the algorithm iteration while moving away from the local optima; therefore, the LFD algorithm continues to explore the search space until finding best solutions for convergence to the trend of them. Notably, F16, F17, F18, F19, F20, F21, and F22 are the last to converge with respect to the LFD algorithm, and it is a fast convergence from the initial steps of the algorithm iterations. Because F19, F20, F21, and F22 are the important and most challenging evolution criteria of this section (composite test functions). Based on the aforementioned results, the LFD algorithm may easily achieve a desired balance between exploration and exploitation hence obtains global optima. Additionally, the numerical results proved the superiority of the LFD algorithm in solving the test optimization problems compared with the competitive algorithms. Note that the main reason behind the high exploration ability of the LFD algorithm is the updating mechanism of positions presented in the mathematical model section through Eqs. (6)–(8). Also, Fig. 4 shows the excellent ability of the LFD algorithm to solve the CEC 2017 suite compared to other competitive algorithms.

4.3.3. Non-parametric statistical test analysis

The Wilcoxon rank-sum test (Wilcoxon, 1945) is a statistical non-parametric test used to examine whether two independent samples are selected from populations having same distribution. It also examines the null hypothesis whether two populations are of the same distribution. The same objective can be utilized to determine if two sets of the obtained solutions are different statistically or not. The result obtained by using the Wilcoxon rank-sum test is a parameter called p-vale, which measures the significance level of algorithms. In this regards, two algorithms must achieve a p-value less than 0.05 to be statistically significant. Table 3 contains the obtained results of p-values at 5% using Wilcoxon test. The LFD algorithm is compared with nine well-known algorithms in the literature: GA and DE, SA, PSO, EHO, MFO, WOA, GOA and HHO. The results show that the proposed algorithm provides very competitive performance and in some cases it outperforms other algorithms. Also, the results in Table 3 confirm the significant superiority of LFD over the other competitor algorithms based on the p-values (< 0.05). Thus, the LFD algorithm is statistical different from these competitor algorithms.

Table 3
Comparison of the proposed LFD algorithm with nine common algorithms based on the Wilcoxon test (for $p \geq 0.05$).

Algorithms	Unimodal functions	Multimodal functions	Composite functions
LFD vs. GA	0.0020	0.0156	0.0029
LFD vs. DE	0.0625	0.0156	9.7656E−04
LFD vs. SA	0.0185	0.2968	4.8828E−04
LFD vs. PSO	0.0136	0.0156	4.8828E−04
LFD vs. EHO	9.7656E−04	0.0156	4.8828E−04
LFD vs. MFO	9.7656E−04	0.1093	0.0014
LFD vs. WOA	0.0185	0.0468	4.8828E−04
LFD vs. GOA	9.7656E−04	0.0156	4.8828E−04
LFD vs. HHO	0.0322	0.2187	4.8828E−04

Table 4
The optimal values obtained by LFD and other competitive algorithms for solving the tension/compression spring design.

Algorithms	Optimum variables			Optimum weight
	d	*D*	*N*	
GA	0.0661	0.6521	5.9446	2.27E−02
DE	0.0500	0.3110	15	1.32E−02
SA	0.0500	0.3173	14.2366	1.29E−02
PSO	0.0514	0.3577	11.6187	1.27E−02
EHO	0.0590	0.5333	6.0819	1.50E−02
MFO	0.0559	0.4688	6.8369	1.30E−02
WOA	0.0587	0.5514	5.0975	1.35E−02
GOA	0.0500	0.3104	15	1.32E−02
HHO	0.0580	0.5295	5.4853	1.33E−02
LFD	0.0517	0.3575	11.2442	1.27E−02

Table 5
Statistical results obtained for the LFD algorithm and other competitive algorithms on solving the tension/compression design problem.

Algorithms	Mean	Std	Best	Worst
GA	2.93E+14	2.87E+14	2.27E−02	7.62E+14
DE	9.86E+12	2.11E+13	1.32E−02	4.76E+13
SA	1.48E−02	2.34E−03	1.29E−02	2.23E−02
PSO	1.27E−02	4.12E−01	1.27E−02	1.28E−02
EHO	1.20E+11	6.34E+11	1.50E−02	3.48E+12
MFO	1.31E−02	1.49E−04	1.30E−02	1.32E−02
WOA	1.39E−02	3.96E−04	1.35E−02	1.43E−02
GOA	1.65E−02	8.57E−03	1.32E−02	5.68E−02
HHO	1.34E−02	1.08E−04	1.33E−02	1.35E−02
LFD	1.38E−02	9.82E−04	1.27E−02	1.56E−02

Table 6
The optimal values obtained by LFD and other competitive algorithms for solving the welded beam design problem.

Algorithms	Optimum variables				Optimum cost
	h	*l*	*t*	*b*	
GA	0.6982	3.0113	4.1993	0.9815	4.99E+00
DE	0.2083	3.6891	8.9814	0.2083	1.77E+00
SA	0.1798	4.1551	8.9987	0.2077	1.78E+00
PSO	0.2157	3.4704	9.0356	0.2658	1.86E+00
EHO	0.1743	8.2809	8.0077	0.2748	2.64E+00
MFO	0.2013	3.2320	10	0.2013	1.81E+00
WOA	0.1974	3.8040	8.7025	0.2261	1.85E+00
GOA	0.1779	3.9145	9.8992	0.2018	1.86E+00
HHO	0.1928	3.9285	9.1378	0.2052	1.78E+00
LFD	0.1857	3.9070	9.1552	0.2051	1.77E+00

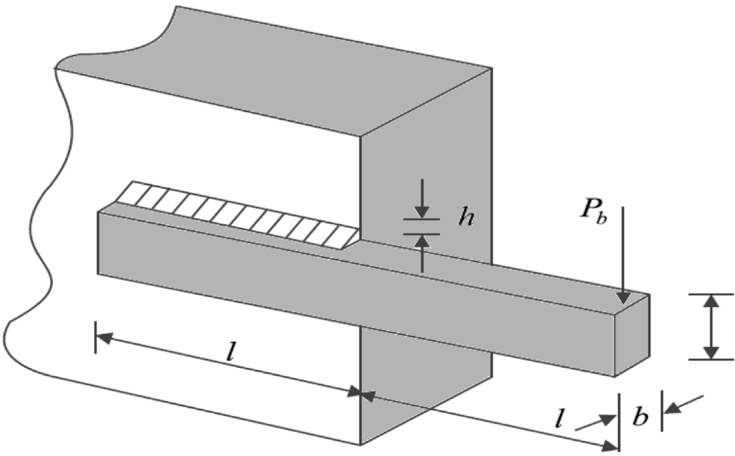


Fig. 6. A schematic illustration for the welded beam problem.

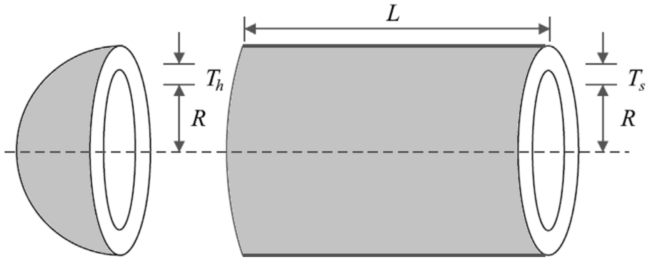


Fig. 7. An illustration for the pressure vessel design problem.

4.4. Experiments on real-world applications

This section is devoted to assess the performance of the LFD algorithm on some real-world engineering applications including tension/compression spring design, welded beam design, and pressure vessel design. Also, the node deployment problem in WSNs is considered for performance evaluation of the LFD algorithm.

4.4.1. Engineering design problems

In spite of the previous findings strongly suggest that LFD can solve real problems, here three real problems from structural design are used to prove the applicability of LFD in solving such engineering design problems with unknown search spaces, which are tension/compression spring design, welded beam design, and pressure vessel design. These problems are used to test and evaluate the LFD algorithm against the competitive algorithms. Accordingly, engineering design problems have different constraints so, a handling method should be applied to handle these various constraints. In the literature, there are several constraint handling methods (Coello, 2002), including penalty functions (e.g., adaptive, annealing, static, dynamic, and death penalty functions). As a consequence, in this work, the death penalty function is utilized to handle different constraints in the engineering design problems. The death penalty function automatically eliminates non-applicable solutions by guiding the metaheuristic algorithms during the optimization process (minimize the large fitness value). The greatest advantages of this method are its low computational cost and its simplicity. The death penalty function is used for handling engineering design constraints. The performance of the LFD algorithm has been compared against seven well-known optimization algorithms, namely GA, DE, SA, PSO, EHO, MFO, WOA, GOA, and HHO.

4.4.1.1. Tension/compression spring design. The main objective of this engineering design problem is to minimize the weight of the tension/compression spring described in Dhiman and Kumar (2017). A schematic illustration for the tension/compression spring problem is shown in Fig. 5. The desirable design must satisfy constraints on the surge frequency, shear stress and deflection. Actually, there are three optimization variables: number of active coils (N), wire diameter (d), and mean coil diameter (D). The tension/compression spring problem can be formulated mathematically as follows

Suppose $\vec{x} = \left[x_1 \;\; x_2 \;\; x_3\right] = [N \;\; d \;\; D]$

Minimize $f(\vec{x}) = \left(x_3 + 2\right) x_2 x_1^2$

Subject to

$$g_1(\vec{x}) = 1 - \frac{x_2^3 x_3}{71785 x_1^4} \leq 0$$

$$g_2(\vec{x}) = \frac{4x_2^2 - x_1 x_2}{12566\left(x_2 x_1^3 - x_1^4\right)} + \frac{1}{5108 x_1^2} \leq 0$$

$$g_3(\vec{x}) = 1 - \frac{140.45 x_1}{x_2^2 x_3} \leq 0$$

$$g_4(\vec{x}) = \frac{x_1 + x_2}{1.5} - 1 \leq 0$$

$$0.05 \leq x_1 \leq 2.00$$

$$0.25 \leq x_2 \leq 1.30$$

$$2.00 \leq x_3 \leq 15.0$$

The obtained optimization results are shown in Table 4, revealing that the LFD algorithm has achieved the best results in solving this problem by minimizing the weight of tension/compression spring. Table 5 illustrates the statistical results obtained for the competitive algorithms over 30 independent runs and 1000 iterations (i.e., maximum).

4.4.1.2. Welded beam design. The fitness function of the welded beam design problem proposed by Coello (2000) minimizes the fabrication cost of the welded beam. That is, the cost of a welded beam design shown in Fig. 6) is minimized keeping in view certain constraints. Optimization constraints in this problem include bending stress (θ) in the beam, the buckling load (Pc), on the shear stress (τ), and the end deflection of the beam (δ). Besides, there are only four optimization variables: length of the clamped bar (l), thickness of the weld (h), height of the bar (t), and thickness of the bar (b). Mathematically, the welded beam design problem is expressed as

Suppose $\vec{x} = \left[x_1 \;\; x_2 \;\; x_3 \;\; x_4\right] = [l \;\; h \;\; t \;\; b]$

Minimize $f(\vec{x}) = 1.10471 x_1^2 x_2 + 0.04811 x_3 x_4 \left(14.0 + x_2\right)$

Subject to:

$$g_1(\vec{x}) = \tau(\vec{x}) - \tau_{\max} \leq 0$$

$$g_2(\vec{x}) = \sigma(\vec{x}) - \sigma_{\max} \leq 0$$

$$g_3(\vec{x}) = \delta(\vec{x}) - \delta_{\max} \leq 0$$

$$g_4(\vec{x}) = x_1 - x_4 \leq 0$$

$$g_5(\vec{x}) = P - P_C(\vec{x}) \leq 0$$

$$g_6(\vec{x}) = 0.125 - x_1 \leq 0$$

$$g_7(\vec{x}) = 1.10471 x_1^2 + 0.04811 x_3 x_4 \left(14.0 + x_2\right) - 5.0 \leq 0$$

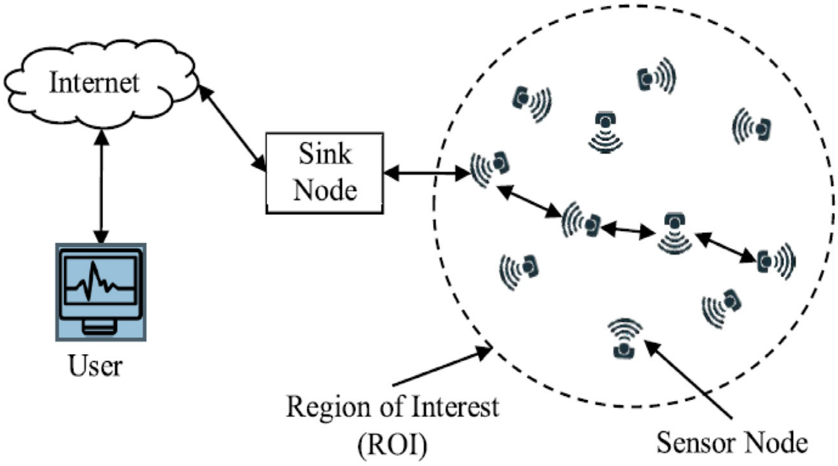


Fig. 8. An illustration for the distribution of sensor nodes in a wireless sensor network.

Table 7
The statistical results obtained for the LFD algorithm and other competitive algorithms on solving the welded beam design problem.

Algorithms	Mean	Std	Best	Worst
GA	3.65E+13	4.45E+13	4.99E+00	8.96E+13
DE	2.64E+00	1.89E+00	1.77E+00	6.01E+00
SA	1.91E+00	1.27E–01	1.78E+00	2.24E+00
PSO	2.36E+00	1.36E–02	1.86E+00	3.54E+00
EHO	3.76E+00	6.79E–01	2.64E+00	4.82E+00
MFO	2.31E+00	4.28E–01	1.81E+00	2.58E+00
WOA	1.93E+00	1.38E–01	1.85E+00	2.09E+00
GOA	2.72E+00	1.37E+00	1.86E+00	8.89E+00
HHO	1.89E+00	1.11E–01	1.78E+00	2.00E+00
LFD	2.30E+00	3.16E–01	1.77E+00	3.04E+00

Table 8
The optimal values obtained by LFD and other competitive algorithms for solving the pressure vessel design problem.

Algorithms	Optimum variables				Optimum cost
	T_s	T_h	R	L	
GA	1.1568	10.4356	52.8657	131.6100	5.88E+04
DE	0.8065	0.3900	40.8686	200	6.20E+03
SA	0.9322	0.4419	45.4545	148.5710	6.73E+03
PSO	1.0574	0.5985	53.3912	74.0986	6.81E+03
EHO	1.1809	0.8893	60.0018	119.5834	1.32E+04
MFO	0.8927	0.4413	46.2585	131.1057	6.11E+03
WOA	0.8359	0.5940	42.6948	169.3842	6.65E+03
GOA	1.2755	2.0300	58.1327	50.3785	1.67E+04
HHO	1.0374	0.5085	53.3112	74.0686	6.51E+03
LFD	0.8777	0.4339	45.4755	139.0654	6.08E+03

Table 9
Statistical results obtained for the LFD algorithm and other competitive algorithms on solving the pressure vessel design problem.

Algorithms	Mean	Std	Best	Worst
GA	2.18E+05	2.25E+05	5.88E+04	3.77E+05
DE	6.36E+03	2.31E+02	6.20E+03	6.53E+03
SA	1.10E+04	6.52E+03	6.73E+03	3.14E+04
PSO	6.96E+03	1.92E+02	6.81E+03	7.03E+03
EHO	8.76E+04	4.62E+04	1.32E+04	1.96E+05
MFO	6.82E+03	1.01E+03	6.11E+03	7.53E+03
WOA	8.32E+03	1.31E+03	6.65E+03	1.10E+04
GOA	1.40E+05	9.56E+04	1.67E+04	3.77E+05
HHO	6.76E+03	1.62E+02	6.51E+03	6.93E+03
LFD	1.60E+04	8.02E+03	6.08E+03	3.62E+04

Table 10
Parameter settings for the coverage area in the WSN.

Parameters	Values
Deployment area	600 ∗ 600 – 800 ∗ 800 – 900 ∗ 900
Number of nodes	100–250
Sensor node model	Mica Mote
Node communication range	100 m
Node sensing range	20 m
Node placement	Uniform
Node energy	Uniform
Max energy	2000 (mA-h)
Threshold (Comfort distance)	$\sqrt{3} * 20$ (Sensing range)

$$0.1 \leq x_1 \leq 2$$
$$0.1 \leq x_2 \leq 10$$
$$0.1 \leq x_3 \leq 10$$
$$0.1 \leq x_4 \leq 2$$

where,

$$\tau(\vec{x}) = \sqrt{(\tau')^2 + 2\tau'\tau''\frac{x_2}{2R} + (\tau'')^2}$$

$$\tau' = \frac{P}{\sqrt{2}x_1x_2}, \quad \tau'' = \frac{MR}{J}, M = P\left(L + \frac{x_2}{2}\right)$$

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

$$J = 2\left\{\sqrt{2}x_1x_2\left[\frac{x_2^2}{4} + \left(\frac{x_1+x_3}{2}\right)^2\right]\right\}$$

$$\sigma(\vec{x}) = \frac{6PL}{x_4x_3^2}, \delta(\vec{x}) = \frac{6PL^3}{Ex_3^2x_4}$$

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

$P = 6000$ lb, $L = 14$ in., $\delta_{\max} = 0.25$ in.
$E = 30 \times 1^6$psi, $G = 12 \times 10^6$ psi.
$\tau_{\max} =$ 13,600 psi, $\sigma_{\max} = 30,000$ psi.

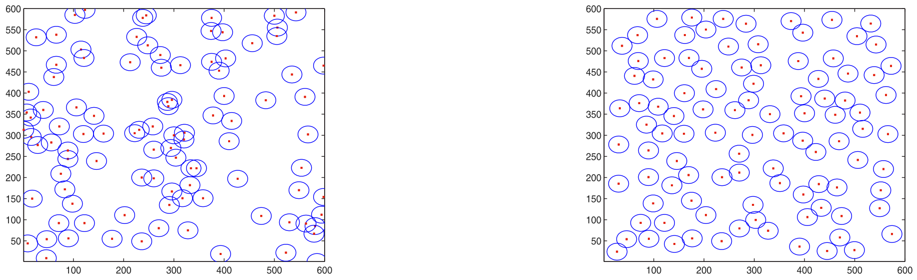
The obtained optimization results reported in Table 6 prove that the LFD algorithm can achieve the best results in solving the welded beam problem by minimizing its weight. The statistical results shown in Table 7 are obtained for the LFD algorithm and the competitive algorithms over 30 independent runs and number of iterations is set to the maximum 1000. it is clear that LFD is capable to find the optimal design with low cost.

4.4.1.3. Pressure vessel design. The fitness function of the pressure vessel design problem (Kannan and Kramer, 1994), as illustrated in Fig. 7, minimizes the total cost (material, forming and welding) of the cylindrical pressure vessel. In this design problem, there are four optimization variables: thickness of the head (T_h), inner radius (R), thickness of the shell (T_s), and length of the cylindrical section without considering the head (L). Mathematically, it is formulated by

Suppose $\vec{x} = \left[x_1 \;\; x_2 \;\; x_3 \;\; x_4\right] = \left[T_h \;\; R \;\; T_s \;\; L\right]$

Minimize $f(\vec{x}) = 0.6224x_1x_3x_4 + 1.7781x_2x_3^2 + 3.1661x_1^2x_4 + 19.84x_1^2x_3$

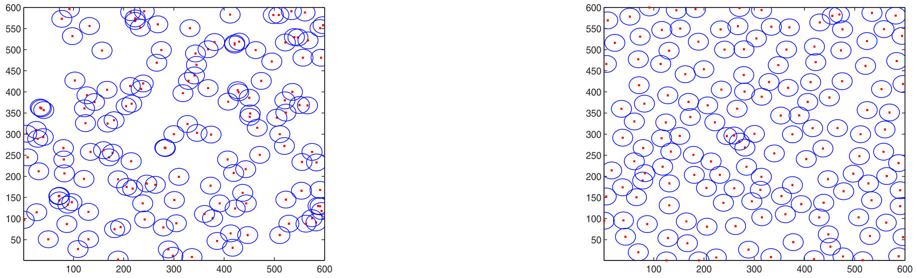
Subject to



(a) Deployment of the A3 algorithm with an overlapping value = 1553.8269, coverage rate = 0.34

(b) Deployment of the LFD algorithm with an overlapping value = 0, coverage rate = 0.35

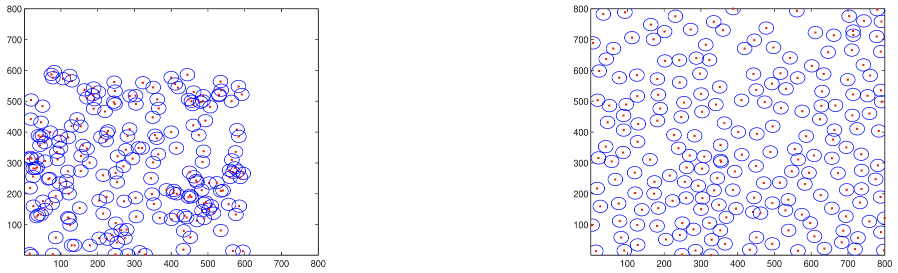
Fig. 9. Deployment of the A3 and LFD algorithms for 100 sensor nodes with 600 ∗ 600 area coverage.



(a) Deployment of the A3 algorithm with an overlapping value = 4214.7066, coverage rate = 0.51

(b) Deployment of the LFD algorithm with an overlapping value = 717.7301, coverage rate = 0.52

Fig. 10. Deployment of the A3 and LFD algorithms for 150 sensor nodes with 600 ∗ 600 area coverage.



(a) Deployment of the A3 algorithm with an overlapping value = 8074.6228, coverage rate = 0.38

(b) Deployment of the LFD algorithm with an overlapping value = 421.0473, coverage rate = 0.39

Fig. 11. Deployment of the A3 and LFD algorithms for 200 sensor nodes with 800 ∗ 800 area coverage.

$$g_1(\vec{x}) = -x_1 + 0.0193x_3 \le 0$$
$$g_2(\vec{x}) = -x_3 + 0.00954x_3 \le 0$$
$$g_3(\vec{x}) = -\pi x_3^2 x_4 - \frac{4}{3}\pi x_3^3 + 1,296,000 \le 0$$
$$g_4(\vec{x}) = x_4 - 240 \le 0$$

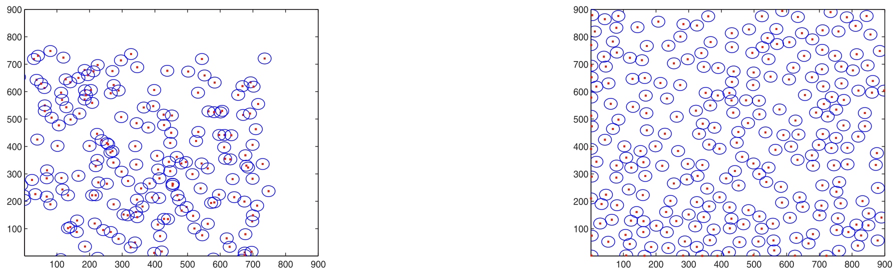
Variable range $0 \le x_1 \le 99$

$$0 \le x_2 \le 99$$
$$10 \le x_3 \le 200$$
$$10 \le x_4 \le 200$$

The obtained optimization results are reported in Table 8, which prove that the LFD algorithm achieved the best results in solving this problem by minimizing the total cost (material, forming and welding) of the cylindrical pressure vessel. Table 9 demonstrates the statistical results obtained for the LFD algorithm and the competitive algorithms over 30 independent runs and the maximum number of iterations is set to be 1000.

4.4.2. Node deployment in wireless sensor networks problem

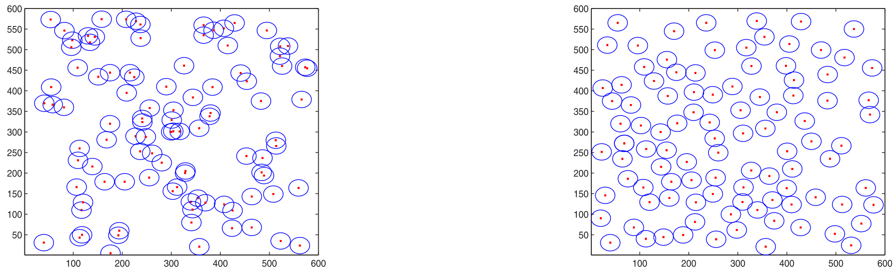
Wireless sensor networks (WSNs) are a collection of sensing nodes with the same sensing/communication range and a single static sink node as shown in Fig. 8. These nodes are distributed randomly in a convex region of interest (ROI) $R = W \times H$, where W and H refer to the width and height of ROI, respectively. One of the most common problems in WSNs is the low area coverage, which appears as a result for random deployment of various topology construction algorithms. This problem can negatively affect the efficiency of networks. Topology control algorithms such as A3, EECDS, and CDS-Rule K are widely used in WSNs to establish and deploy network nodes. The main merits



(a) Deployment of the A3 algorithm with an overlapping value = 4820.5871, coverage rate = 0.38

(b) Deployment of the LFD algorithm with an overlapping value = 585.2087, coverage rate = 0.39

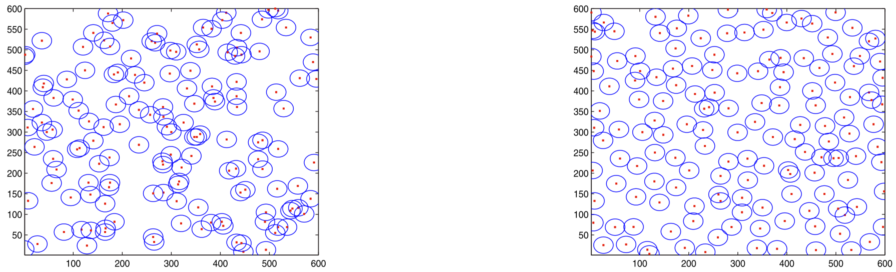
Fig. 12. Deployment of the A3 and LFD algorithms for 250 sensor nodes with 900 ∗ 900 area coverage.



(a) Deployment of the EECDS algorithm with an overlapping value = 1956.1759, coverage rate = 0.34

(b) Deployment of the LFD algorithm with an overlapping value = 0, coverage rate = 0.35

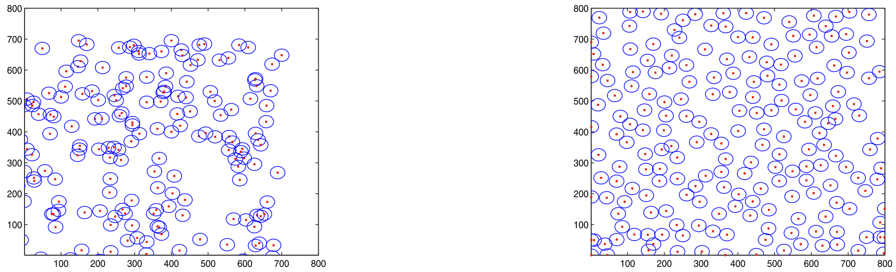
Fig. 13. Deployment of the EECDS and LFD algorithms for 100 sensor nodes with 600 ∗ 600 area coverage.



(a) Deployment of the EECDS algorithm with an overlapping value = 3717.5156, coverage rate = 0.51

(b) Deployment of the LFD algorithm with an overlapping value = 1070.833, coverage rate = 0.52

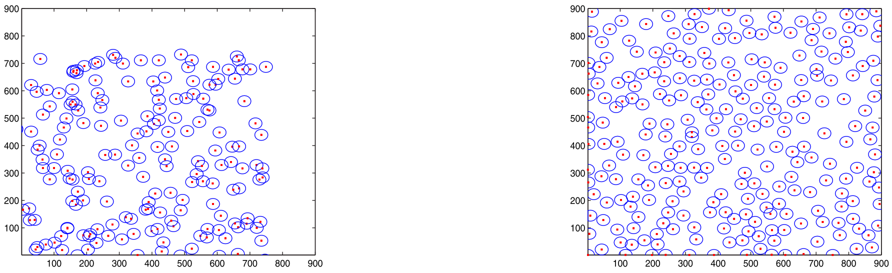
Fig. 14. Deployment of the EECDS and LFD algorithms for 150 sensor nodes with 600 ∗ 600 area coverage.



(a) Deployment of the EECDS algorithm with an overlapping value = 4084.9335, coverage rate = 0.38

(b) Deployment of the LFD algorithm with an overlapping value = 534.31, coverage rate = 0.39

Fig. 15. Deployment of the EECDS and LFD algorithms for 200 sensor nodes with 800 ∗ 800 area coverage.



(a) Deployment of the EECDS algorithm with an overlapping value = 4399.9913,
coverage rate = 0.38

(b) Deployment of the LFD algorithm with an overlapping value = 657.1558,
coverage rate = 0.39

Fig. 16. Deployment of the EECDS and LFD algorithms for 250 sensor nodes with 900 $*$ 900 area coverage.

of a good topology control algorithm are saving energy, reducing the inference between sensor nodes, extending lifetime of networks, and providing connected topology.

In the problem of maximizing area coverage in WSNs, given a number of sensors within same sensing ranges in 2D domain R, the main objective is to determine locations of all sensors with the greatest possible (i.e., maximum) area of coverage on R. This problem is confirmed to be NP-hard (Yoon and Kim, 2013). It is formulated as

Input: Number of sensors n, sensing radius of a sensor r_i, and width W, length H of the 2D domain R.
Output: 2D coordinates of all n sensors.
Objective: Area coverage (coR) of n sensors on R that maximizes

$$\mathrm{coR} = \mathrm{area}\left(\bigcup_{i=1}^{n} c_{r_i}\left(\mathrm{x}_i, \mathrm{y}_i\right) \cap R\right) \rightarrow \max \tag{20}$$

where $c_{r_i}\left(x_i, y_i\right)$ is a circle centred at $\left(x_i, y_i\right)$ and the radius is r_i; the $area(X)$ is the area of the domain X.

In this work, the LFD algorithm is utilized in WSNs to solve the area coverage problem and minimize the overlapping of sensor nodes, which made by topology construction algorithms: A3, EECDS, and CDS-Rule K algorithms. The Atarraya simulator (Wightman and Labrador, 2009) is utilized to create the test network dataset used in the experiments. The obtained results show that the LFD algorithm has a higher capability to minimize graphical and computational overlap of the sensor nodes than the A3, EECDS, and CDS-Rule K algorithms. The parameters settings of this experiment are given in Table 10.

The objective function of the proposed LFD algorithm is redefined for simulation of the area coverage to minimize the overlapping of sensor nodes in the WSNs. This function is called the overlapping fitness function *(Olap)* with complexity of $O(n^2)$ and it is defined by

$$\mathrm{Olap}(S) = \sum_{i=1}^{n}\left(\sum_{j=i+1}^{n} \mathrm{overlap}\left(x_i, x_j\right) + \sum_{m=1}^{4} \mathrm{overlap}\left(x_i, b_m\right)\right) \tag{21}$$

where S are the sensor nodes in the area coverage, $overlap(x_i, x_j)$ is the overlapping value between ith and jth sensor nodes, and $overlap(x_i, b_m)$ is the overlapping value between ith sensor nodes and the boundaries of the area coverage represented by b_m, which represents the sensing range that is outside the area coverage.

In this context, the simulation experiments on WSNs are performed as follows: firstly, generating the numerical and graphical datasets of WSNs with network sizes equal 100, 150, 250, and 250 from the Atarraya simulator for A3/EECDS/CDS-Rule K topology construction algorithms. Secondly, initializing the LFD algorithm to read these datasets and identifying a threshold value which is often equal to $\sqrt{3} \times NodeSensingRange$ (Liao et al., 2011). Thirdly, making the LFD algorithm working on Eq. (21) as a fitness function. Finally, running the LFD algorithm 100 iterations over the four different network sizes for each algorithm (A3/EECDS/CDS-Rule K) separately. Subsequently, generating the optimized results in terms of overlapping value and coverage ratio for each algorithm using the LFD algorithm. The LFD algorithm and A3/EECDS/CDS-Rule K topology construction algorithms perform 100 iterations over the four different network sizes to obtain satisfactory results. The obtained results show that the LFD algorithm outperforms the A3/EECDS/CDS-Rule K topology construction algorithms in the distributing area in some ROIs with the lowest value of overlap as shown in Figs. 9 to 12, and Figs. 13 to 16, and Figs. 17 to 20. It is clear from the results shown in Figs. 9 to 20, compared with the A3/EECDS/CDS-Rule K algorithms, the LFD algorithm achieved high performance in deploying sensor nodes with a low amount of overlap.

Also, the total area covered by nodes adopts the union concept. Always, the value of coverage rate ought be smaller or equal to 1. The coverage rate C_r is calculated using

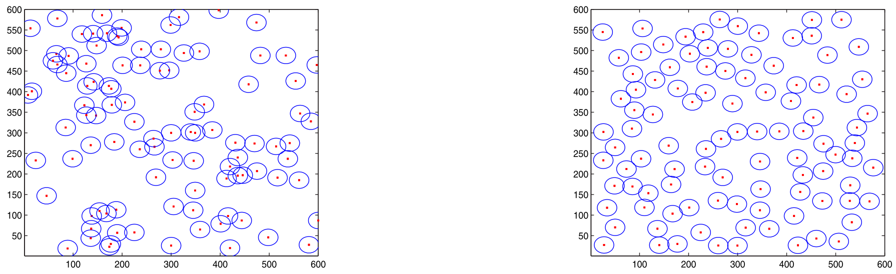
$$C_r = \frac{\bigcup_{i=1\dots n} A_i}{A} \tag{22}$$

where, A_i refers to the covered area by the S_i node, n refers to the number of mobile sensor nodes, and A is the total area of service. Computing of coverage ratio is based on the Grid scan method (Shen et al., 2006; Chen et al., 2007).

Figs. 9, 13, and 17 show that the interference values obtained by the A3/EECDS/CDS-Rule K algorithms are 1553.8269/1956.1759/1714.3404, respectively while the value obtained by the LFD algorithm reflects the efficiency of the LFD algorithm in distributing sensor nodes with minimal node overlapping values. Figs. 10, 11, 12, 14, 15, 16, 18, 19, and 20 prove the merits of the LFD algorithm in minimizing: (1) overlapping nodes, (2) redundant areas, (3) negative effect of the blind subareas, where there is not an adequate number of sensor nodes to cover these subareas. Furthermore, the lFD algorithm can maximize the coverage ratio. This effect is the troubling problem of redundancy with their neighbours having less sensors in the same area coverage as depicted in Figs. 11, 12, 15, 16, 19, and 20. Therefore, the LFD algorithm is regarded as a new promising algorithm in the field of WSNs.

5. Conclusions

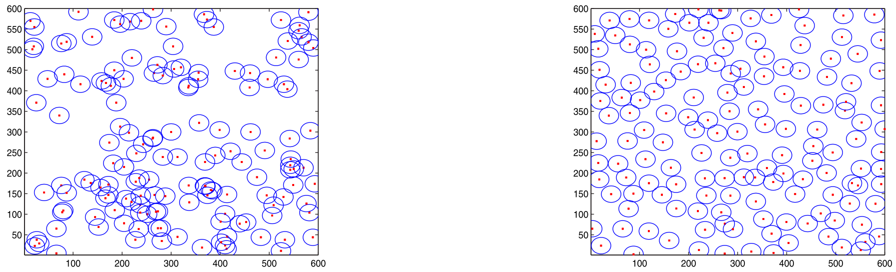
In this paper, a Lévy flight distribution based algorithm is introduced and assessed on a number of well-known test bed optimization problems: the CEC2017 benchmark functions and three engineering optimization problems, including tension/compression, welded beam, and pressure vessel design problems. Inspecting the results of the LFD algorithm on the CEC 2017 suite and three engineering problems, it is evident that the LFD algorithm can handle efficiently difficulties of a constrained search space compared to other well-known algorithms, such as SA, GOA, GA, DE, WOA, PSO, EHO, MFO, and HHO. In addition, the node deployment problem in wireless sensor networks with different network sizes is solved, where LFD provides a better deployment schema than A3, energy efficient connected dominating set,



(a) Deployment of the CDS-Rule K algorithm with an overlapping value = 1714.3404,
coverage rate = 0.34

(b) Deployment of the LFD algorithm with an overlapping value = 0,
coverage rate = 0.35

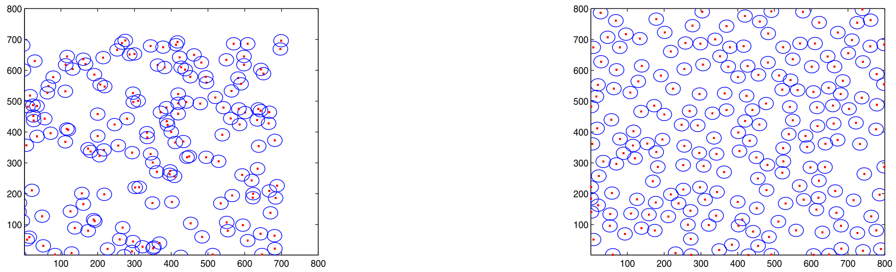
Fig. 17. Deployment of the CDS-Rule K and LFD algorithms for 100 sensor nodes with 600 ∗ 600 area coverage.



(a) Deployment of the CDS-Rule K algorithm with an overlapping value = 4568.1052,
coverage rate = 0.51

(b) Deployment of the LFD algorithm with an overlapping value = 714.7285,
coverage rate = 0.53

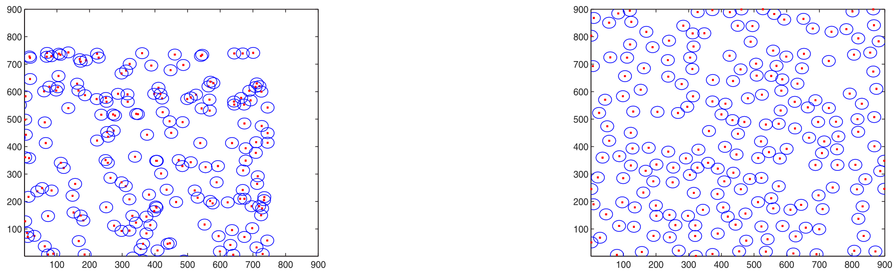
Fig. 18. Deployment of the CDS-Rule K and LFD algorithms for 150 sensor nodes with 600 ∗ 600 area coverage.



(a) Deployment of the CDS-Rule K algorithm with an overlapping value = 3647.1801,
coverage rate = 0.38

(b) Deployment of the LFD algorithm with an overlapping value = 493.0435,
coverage rate = 0.39

Fig. 19. Deployment of the CDS-Rule *K* and LFD algorithms for 200 sensor nodes with 800 ∗ 800 area coverage.



(a) Deployment of the CDS-Rule K algorithm with an overlapping value = 5336.0905,
coverage rate = 0.38

(b) Deployment of the LFD algorithm with an overlapping value = 320.3862,
coverage rate = 0.40

Fig. 20. Deployment of the CDS-Rule K and LFD algorithms for 250 sensor nodes with 900 ∗ 900 area coverage.

and CDS-Rule K topology construction algorithms. The optimization problems considered in the experiments are successfully solved, which demonstrates the practical merits of the proposed LFD algorithm over other metaheuristic algorithms and A3/EECDS/CDS-Rule K topology construction algorithms in case of statistical test and coverage ratio. In summary, the discussion and findings of this work clearly demonstrate the quality of the exploration, exploitation, local optima avoidance, and convergence rate of the LFD algorithm. Additionally, all reported results revealed that the LFD algorithm can solve different optimization problems. Evaluating the performance of the LFD algorithm on other real-world optimization problems especially from other fields (e.g., data mining and image processing) is important, and can thereby be contemplated as future directions. Enhancement of time complexity and the computational cost of the LFD algorithm is a good future work for solving discrete and multi-objective problems. Moreover, inclusion of other techniques within the search process is also worthy of attention.

CRediT authorship contribution statement

Essam H. Houssein: Supervision, Conceptualization, Methodology, Writing - original draft. **Mohammed R. Saad:** Software, Writing - original draft. **Fatma A. Hashim:** Software, Resources. **Hassan Shaban:** Supervision, Investigation. **M. Hassaballah:** Supervision, Visualization, Writing - review & editing.

Declaration of competing interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

Acknowledgements

The authors would like to thank Professor Ajith Abraham, the Associate Editor, and the six anonymous reviewers for their helpful and constructive comments, which considerably improved the quality of the paper.

References

Ab Wahab, M.N., Nefti-Meziani, S., Atyabi, A., 2015. A comprehensive review of swarm optimization algorithms. PLoS One 10 (5), e0122827.

Ahmed, M.M., Houssein, E.H., Hassanien, A.E., Taha, A., Hassanien, E., 2019. Maximizing lifetime of large-scale wireless sensor networks using multi-objective whale optimization algorithm. Telecommun. Syst. 1–17.

Ali, M.Z., Awad, N.H., Reynolds, R.G., Suganthan, P.N., 2018. A balanced fuzzy cultural algorithm with a modified Lévy flight search for real parameter optimization. Inform. Sci. 447, 12–35.

Amoretti, M., 2014. Evolutionary strategies for ultra-large-scale autonomic systems. Inform. Sci. 274, 1–16.

Back, T., 1996. Evolutionary Algorithms in Theory and Practice: Evolution Strategies, Evolutionary Programming, Genetic Algorithms. Oxford university press.

Binh, H.T.T., Hanh, N.T., Dey, N., et al., 2018. Improved cuckoo search and chaotic flower pollination optimization algorithm for maximizing area coverage in wireless sensor networks. Neural Comput. Appl. 30 (7), 2305–2317.

Bonabeau, E., Marco, D.d.R.D.F., Dorigo, M., Théraulaz, G., Theraulaz, G., et al., 1999. Swarm Intelligence: FrOm Natural to Artificial Systems. 1. Oxford university press.

Boussaïd, I., Lepagnot, J., Siarry, P., 2013. A survey on optimization metaheuristics. Inform. Sci. 237, 82–117.

Çelik, E., 2020. A powerful variant of symbiotic organisms search algorithm for global optimization. Eng. Appl. Artif. Intell. 87, 103294.

Chen, J., Li, S., Sun, Y., 2007. Novel deployment schemes for mobile sensor networks. Sensors 7 (11), 2907–2919.

Cheraghalipour, A., Hajiaghaei-Keshteli, M., Paydar, M.M., 2018. Tree growth algorithm (TGA): A novel approach for solving optimization problems. Eng. Appl. Artif. Intell. 72, 393–414.

Coello, C.A.C., 2000. Use of a self-adaptive penalty approach for engineering optimization problems. Comput. Ind. 41 (2), 113–127.

Coello, C.A.C., 2002. Theoretical and numerical constraint-handling techniques used with evolutionary algorithms: A survey of the state of the art. Comput. Methods Appl. Mech. Engrg. 191 (11–12), 1245–1287.

Das, S., Suganthan, P.N., 2010. Differential evolution: A survey of the state-of-the-art. IEEE Trans. Evol. Comput. 15 (1), 4–31.

Dhiman, G., Kumar, V., 2017. Spotted hyena optimizer: A novel bio-inspired based metaheuristic technique for engineering applications. Adv. Eng. Softw. 114, 48–70.

Dinkar, S.K., Deep, K., 2018. An efficient opposition based Lévy flight antlion optimizer for optimization problems. J. Comput. Sci. 29, 119–141.

Dorigo, M., Birattari, M., Stutzle, T., 2006. Ant colony optimization. IEEE Comput. Intell. Mag. 1 (4), 28–39.

Eberhart, R., Kennedy, J., 1995a. A new optimizer using particle swarm theory. In: International Symposium on Micro Machine and Human Science. IEEE, pp. 39–43.

Eberhart, R., Kennedy, J., 1995b. Particle swarm optimization. In: IEEE International Conference on Neural Networks, Vol. 4. Citeseer, pp. 1942–1948.

Emary, E., Zawbaa, H.M., 2019. Feature selection via Lèvy antlion optimization. Pattern Anal. Appl. 22 (3), 857–876.

Emary, E., Zawbaa, H.M., Sharawi, M., 2019. Impact of Lèvy flight on modern meta-heuristic optimizers. Appl. Soft Comput. 75, 775–789.

Etminaniesfahani, A., Ghanbarzadeh, A., Marashi, Z., 2018. Fibonacci indicator algorithm: A novel tool for complex optimization problems. Eng. Appl. Artif. Intell. 74, 1–9.

Faramarzi, A., Heidarinejad, M., Stephens, B., Mirjalili, S., 2020. Equilibrium optimizer: A novel optimization algorithm. Knowl.-Based Syst. 191, 105190.

Guo, Z., Yue, X., Yang, H., Liu, K., Liu, X., 2017. Enhancing social emotional optimization algorithm using local search. Soft Comput. 21 (24), 7393–7404.

Gupta, S., Deep, K., 2019. A novel random walk grey wolf optimizer. Swarm Evol. Comput. 44, 101–112.

Gupta, G.P., Jha, S., 2018. Integrated clustering and routing protocol for wireless sensor networks using cuckoo and harmony search based metaheuristic techniques. Eng. Appl. Artif. Intell. 68, 101–109.

Hashim, F.A., Houssein, E.H., Mabrouk, M.S., Al-Atabany, W., Mirjalili, S., 2019. Henry gas solubility optimization: A novel physics-based algorithm. Future Gener. Comput. Syst. 101, 646–667.

Hayyolalam, V., Kazem, A.A.P., 2020. Black widow optimization algorithm: A novel meta-heuristic approach for solving engineering optimization problems. Eng. Appl. Artif. Intell. 87, 103249.

Heidari, A.A., Mirjalili, S., Faris, H., Aljarah, I., Mafarja, M., Chen, H., 2019. Harris hawks optimization: Algorithm and applications. Future Gener. Comput. Syst. 97, 849–872.

Heidari, A.A., Pahlavani, P., 2017. An efficient modified grey wolf optimizer with Lévy flight for optimization tasks. Appl. Soft Comput. 60, 115–134.

Holland, J.H., 1992. Genetic algorithms. Sci. Am. 267 (1), 66–73.

Houssein, E.H., Hosney, M.E., Oliva, D., Mohamed, W.M., Hassaballah, M., 2020a. A novel hybrid Harris hawks optimization and support vector machines for drug design and discovery. Comput. Chem. Eng. 133, 106656.

Houssein, E.H., Saad, M.R., Hussain, K., Zhu, W., Shaban, H., Hassaballah, M., 2020b. Optimal sink node placement in large scale wireless sensor networks based on harris' hawk optimization algorithm. IEEE Access 8, 19381–19397.

Hussain, K., Salleh, M.N.M., Cheng, S., Shi, Y., 2019. Metaheuristic research: A comprehensive survey. Artif. Intell. Rev. 52 (4), 2191–2233.

Javidy, B., Hatamlou, A., Mirjalili, S., 2015. Ions motion algorithm for solving optimization problems. Appl. Soft Comput. 32, 72–79.

Kannan, B., Kramer, S.N., 1994. An augmented lagrange multiplier based method for mixed integer discrete continuous optimization and its applications to mechanical design. J. Mech. Des. 116 (2), 405–411.

Karaboga, D., Gorkemli, B., Ozturk, C., Karaboga, N., 2014. A comprehensive survey: Artificial bee colony (ABC) algorithm and applications. Artif. Intell. Rev. 42 (1), 21–57.

Kaur, S., Awasthi, L.K., Sangal, A., Dhiman, G., 2020. Tunicate swarm algorithm: A new bio-inspired based metaheuristic paradigm for global optimization. Eng. Appl. Artif. Intell. 90, 103541.

Kaveh, A., Share, M.A., Moslehi, M., 2013. Magnetic charged system search: A new meta-heuristic algorithm for optimization. Acta Mech. 224 (1), 85–107.

Krishnanand, K., Ghose, D., 2009. Glowworm swarm optimization for simultaneous capture of multiple local optima of multimodal functions. Swarm Intell. 3 (2), 87–124.

Langdon, W.B., Poli, R., 2013. Foundations of Genetic Programming. Springer Science & Business Media.

LaTorre, A., Muelas, S., Peña, J.-M., 2015. A comprehensive comparison of large scale global optimizers. Inform. Sci. 316, 517–549.

Liao, W.-H., Kao, Y., Li, Y.-S., 2011. A sensor deployment approach using glowworm swarm optimization algorithm in wireless sensor networks. Expert Syst. Appl. 38 (10), 12180–12188.

Ling, Y., Zhou, Y., Luo, Q., 2017. Lévy flight trajectory-based whale optimization algorithm for global optimization. IEEE Access 5, 6168–6186.

Liu, W., Wang, Z., Liu, X., Zeng, N., Bell, D., 2018. A novel particle swarm optimization approach for patient clustering from emergency departments. IEEE Trans. Evol. Comput. 23 (4), 632–644.

Ly, D.T.H., Hanh, N.T., Binh, H.T.T., Nghia, N.D., 2015. An improved genetic algorithm for maximizing area coverage in wireless sensor networks. In: International Symposium on Information and Communication Technology. ACM, pp. 61–66.

Magdziarz, M., Szczotka, W., 2016. Quenched trap model for Lévy flights. Commun. Nonlinear Sci. Numer. Simul. 30 (1–3), 5–14.

Mann, P.S., Singh, S., 2017. Improved metaheuristic based energy-efficient clustering protocol for wireless sensor networks. Eng. Appl. Artif. Intell. 57, 142–152.

Marler, R., Arora, J.S., 2004. Survey of multi-objective optimization methods for engineering. Struct. Multidiscip. Optim. 26 (6), 369–395.

Mirjalili, S., 2015. Moth-flame optimization algorithm: A novel nature-inspired heuristic paradigm. Knowl.-Based Syst. 89, 228–249.

Mirjalili, S., 2016. Dragonfly algorithm: A new meta-heuristic optimization technique for solving single-objective, discrete, and multi-objective problems. Neural Comput. Appl. 27 (4), 1053–1073.

Mirjalili, S., Lewis, A., 2016. The whale optimization algorithm. Adv. Eng. Softw. 95, 51–67.

Moghdani, R., Salimifard, K., 2018. Volleyball premier league algorithm. Appl. Soft Comput. 64, 161–185.

Mortazavi, A., Toğan, V., Nuhoğlu, A., 2018. Interactive search algorithm: A new hybrid metaheuristic optimization algorithm. Eng. Appl. Artif. Intell. 71, 275–292.

Neshat, M., Sepidnam, G., Sargolzaei, M., Toosi, A.N., 2014. Artificial fish swarm algorithm: A survey of the state-of-the-art, hybridization, combinatorial and indicative applications. Artif. Intell. Rev. 42 (4), 965–997.

Ojha, V.K., Abraham, A., Snášel, V., 2017. Metaheuristic design of feedforward neural networks: A review of two decades of research. Eng. Appl. Artif. Intell. 60, 97–116.

Okulewicz, M., Mańdziuk, J., 2019. A metaheuristic approach to solve dynamic vehicle routing problem in continuous search space. Swarm Evol. Comput. 48, 44–61.

Ouyang, H.-b., Gao, L.-q., Kong, X.-y., Zou, D.-x., Li, S., 2015. Teaching-learning based optimization with global crossover for global optimization problems. Appl. Math. Comput. 265, 533–556.

Ramezani, F., Lotfi, S., 2013. Social-based algorithm (SBA). Appl. Soft Comput. 13 (5), 2837–2856.

Rashedi, E., Nezamabadi-Pour, H., Saryazdi, S., 2009. GSA: A gravitational search algorithm. Inform. Sci. 179 (13), 2232–2248.

Saremi, S., Mirjalili, S., Lewis, A., 2017. Grasshopper optimisation algorithm: Theory and application. Adv. Eng. Softw. 105, 30–47.

Shadravan, S., Naji, H., Bardsiri, V.K., 2019. The sailfish optimizer: A novel nature-inspired metaheuristic algorithm for solving constrained engineering optimization problems. Eng. Appl. Artif. Intell. 80, 20–34.

Shen, X., Chen, J., Sun, Y., 2006. Grid scan: A simple and effective approach for coverage issue in wireless sensor networks. In: IEEE International Conference on Communications, Vol. 8. IEEE, pp. 3480–3484.

Simon, D., 2013. Evolutionary Optimization Algorithms. John Wiley & Sons.

Simpson, A.R., Dandy, G.C., Murphy, L.J., 1994. Genetic algorithms compared to other techniques for pipe optimization. J. Water Resour. Plan. Manage. 120 (4), 423–443.

Sulaiman, M.H., Mustaffa, Z., Saari, M.M., Daniyal, H., 2020. Barnacles mating optimizer: A new bio-inspired algorithm for solving engineering optimization problems. Eng. Appl. Artif. Intell. 87, 103330.

Tang, K.-S., Man, K.-F., Kwong, S., He, Q., 1996. Genetic algorithms and their applications. IEEE Signal Process. Mag. 13 (6), 22–37.

Truong, T.K., Li, K., Xu, Y., 2013. Chemical reaction optimization with greedy strategy for the 0–1 knapsack problem. Appl. Soft Comput. 13 (4), 1774–1780.

Van Laarhoven, P.J., Aarts, E.H., 1987. Simulated annealing. In: Simulated Annealing: Theory and Applications. Springer, pp. 7–15.

Wang, G.-G., Deb, S., Coelho, L.d.S., 2015. Elephant herding optimization. In: International Symposium on Computational and Business Intelligence. IEEE, pp. 1–5.

Wightman, P.M., Labrador, M.A., 2008. A3: A topology construction algorithm for wireless sensor networks. In: IEEE Global Telecommunications Conference. IEEE, pp. 1–6.

Wightman, P.M., Labrador, M.A., 2009. Atarraya: A simulation tool to teach and research topology control algorithms for wireless sensor networks. In: International Conference on Simulation Tools and Techniques. p. 26.

Wilcoxon, F., 1945. Individual comparisons by ranking methods. Biom. Bull. 1 (6), 80–83.

Wolpert, D., 1997. No free lunch theorem for optimization. IEEE Trans. Evol. Comput. 1 (1), 467–482.

Wu, J., Cardei, M., Dai, F., Yang, S., 2006. Extended dominating set and its applications in ad hoc networks using cooperative communication. IEEE Trans. Parallel Distrib. Syst. 17 (8), 851–864.

Wu, G., Mallipeddi, R., Suganthan, P., 2017. Problem Definitions and Evaluation Criteria for the CEC2017 Competition on Constrained Real-Parameter Optimization. Technical Report, Nanyang Technological University, Singapore, URL http://www.ntu.edu.sg/home/EPNSugan/index_files/CEC2017.

Xu, Y., Chen, H., Heidari, A.A., Luo, J., Zhang, Q., Zhao, X., Li, C., 2019. An efficient chaotic mutative moth-flame-inspired optimizer for global optimization tasks. Expert Syst. Appl. 129, 135–155.

Yang, X.-S., Cui, Z., Xiao, R., Gandomi, A.H., Karamanoglu, M., 2013. Swarm Intelligence and Bio-Inspired Computation: Theory and Applications. Newnes.

Yang, X.-S., Deb, S., Zhao, Y.-X., Fong, S., He, X., 2018. Swarm intelligence: Past, present and future. Soft Comput. 22 (18), 5923–5933.

Yiyue, W., Hongmei, L., Hengyang, H., 2012. Wireless sensor network deployment using an optimized artificial fish swarm algorithm. In: International Conference on Computer Science and Electronics Engineering, Vol. 2. IEEE, pp. 90–94.

Yoon, Y., Kim, Y.-H., 2013. An efficient genetic algorithm for maximum coverage deployment in wireless sensor networks. IEEE Trans. Cybern. 43 (5), 1473–1483.

Yuanyuan, Z., Jia, X., Yanxiang, H., 2006. Energy efficient distributed connected dominating sets construction in wireless sensor networks. In: International Conference on Wireless Communications and Mobile Computing. ACM, pp. 797–802.

Zhang, H., Xie, J., Hu, Q., Shao, L., Chen, T., 2018. A hybrid DPSO with Lévy flight for scheduling MIMO radar tasks. Appl. Soft Comput. 71, 242–254.

Zhao, W., Wang, L., Zhang, Z., 2019. Atom search optimization and its application to solve a hydrogeologic parameter estimation problem. Knowl.-Based Syst. 163, 283–304.