Research Article

Amol Adamuthe* and Abdulhameed Pathan

Enhanced shuffled frog leaping algorithm with improved local exploration and energy-biased load reduction phase for load balancing of gateways in WSNs

https://doi.org/10.1515/comp-2020-0218
received August 14, 2020; accepted December 01, 2020

Abstract: Wireless sensor networks (WSNs) have grown widely due to their application in various domains, such as surveillance, healthcare, telecommunication, etc. In WSNs, there is a necessity to design energy-efficient algorithms for different purposes. Load balancing of gateways in cluster-based WSNs is necessary to maximize the lifetime of a network. Shuffled frog leaping algorithm (SFLA) is a popular heuristic algorithm that incorporates a deterministic approach. Performance of any heuristic algorithm depends on its exploration and exploitation capability. The main contribution of this article is an enhanced SFLA with improved local search capability. Three strategies are tested to enhance the local search capability of SFLA to improve the load balancing of gateways in WSNs. The first proposed approach is deterministic in which the participation of the global best solution in information exchange is increased. The next two variations reduces the deterministic approach in the local search component of SFLA by introducing probability-based selection of frogs for information exchange. All three strategies improved the success of local search. Second contribution of article is increased lifetime of gateways in WSNs with a novel energy-biased load reduction phase introduced after the information exchange step. The proposed algorithm is tested with 15 datasets of varying areas of deployment, number of sensors and number of gateways. Proposed ESFLA-RW variation shows significant improvement over other variations in terms of successful local explorations, best fitness values, average fitness values and convergence rate for all datasets. Obtained results of proposed ESFLA-RW are significantly better in terms of network energy consumption, load balancing, first gateway die and network life. The proposed variations are tested to check the effect of various algorithm-specific parameters namely frog population size, probability of information exchange and probability of energy-biased load reduction phase. Higher population size and probabilities give better solutions and convergence rate.

Keywords: shuffled frog leaping algorithm, load balancing of gateways, wireless sensor networks, evolutionary algorithms, roulette wheel selection, stochastic universal sampling

1 Introduction

Wireless sensor networks (WSNs) play a very crucial role in today's growing technological era. WSN applications are present in different areas, such as environmental monitoring [1], defense [2], healthcare [3], automation [4] and farming [5]. WSNs have become an important aspect of information transfer these days. The design and development in technology have made it feasible to develop tiny sensor nodes that can get connected to WSNs easily and perform the task of sensing and transmission of data. WSNs consist of sensor nodes that are responsible for sensing data, transmission of data to base stations or other sensor nodes. Energy is required to perform these operations in WSNs [6,7]. The only source of energy to sensor nodes is battery power. One major problem in these types of sensors is they come with very limited power sources and their batteries cannot be either replaced or recharged easily [8]. This leads to the need for an energy-efficient data transmission plan, which is a very challenging and important task.

* **Corresponding author: Amol Adamuthe,** Department of CS & IT, RIT Rajaramnagar, Maharashtra, India,
e-mail: amol.admuthe@gmail.com
Abdulhameed Pathan: Department of CSE, RIT Rajaramnagar, Maharashtra, India, e-mail: abdulpathan1994@gmail.com

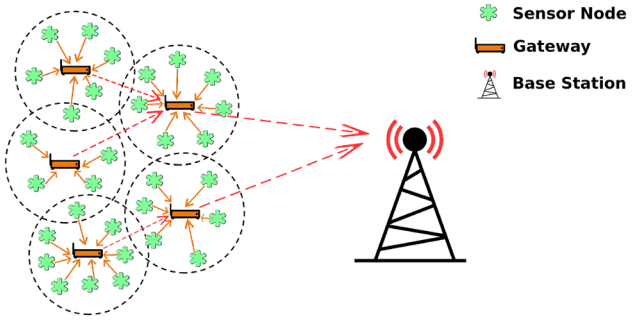

Figure 1: Communication of WSNs.

In cluster-based WSNs, a cluster head (CH) or gateway is responsible to collect and process data received from sensor nodes in the cluster region. Processed data are then transferred to the base station or another CH depending on its routing architecture [9]. Usually, CHs are deployed with high-powered batteries compared to other sensor nodes that are deployed with low-powered batteries [10]. The process of WSNs, communication is shown in Figure 1. Gateways are operated by batteries and have limited power sources. Improper load balancing can result in the early death of gateways due to excess consumption of energy, which could increase the load on nearby gateways and the death chain will be continued. There is a need for proper load balancing of gateways to avoid such problems [11].

In Figure 2, there are three gateways named g_1, g_2 and g_3 with a cluster of 6, 4 and 2 sensor nodes, respectively. Gateway g_1 has more load than other gateways, which will lead to the early death of g_1. This problem is solved by proper balancing of load on gateways as shown in Figure 2. In the literature, different algorithms are presented for optimization of WSNs [12–17]. Selection of appropriate node to become CH is presented in ref. [18], load balancing of gateways by assigning best feasible sensor nodes to gateways is investigated in refs. [19,20] and selection of best routing algorithms to send data from gateways to base station due to transmission range limitations is presented in ref. [21]. Allocation of 'n' gateways

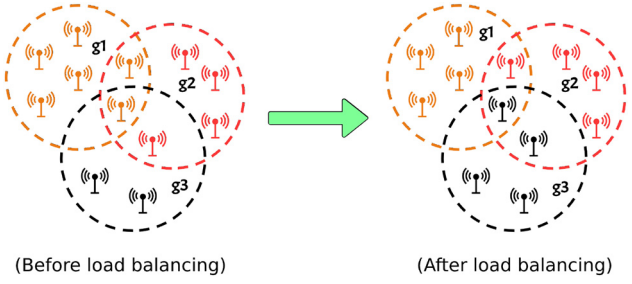

Figure 2: Example of gateways load balancing.

to 'm' sensor nodes is combinatorial problem with ${}^{m}C_{n}$ combination. The combinations increase exponentially with the increase in the number of sensor nodes and gateways. Huge combinations make the load balancing a complex problem ref. [22].

Nature-inspired algorithms are investigated to solve different NP problems in different domains, such as healthcare, image processing, civil engineering, mechanical engineering, science, etc. Evolutionary algorithms are investigated in prediction [23–25], healthcare [26], computer science [27], feature selection problem [28,29], sentiment analysis [30] and turbines layout optimization problems [31].

In the literature, different nature-inspired and bio-inspired techniques are investigated for clustering problems in WSNs [10,13,19,32,33].

Nature-inspired algorithms are inspired by some natural phenomena. In recent years, a wide range of problems are investigated using nature-inspired algorithm, such as ray optimization, gravitational search algorithm, harmony search algorithm, flower pollination algorithm, cuckoo search and bat algorithm.

Shuffled frog leaping algorithm (SFLA) is a population-based heuristic algorithm that imitates the behavior of a frog population searching for the food location [34]. SFLA was built on the idea of evolving individual memes and exchanging information with other memes. Each individual meme represents its thoughts, cultural information and behavior that can be exchanged with global memes to improve overall knowledge. Thus, SFLA takes the benefits of memetic algorithm and particle swarm optimization [35]. Memes in SFLA spread faster than genes in GA resulting in a better convergence rate [34]. The self evolutionary approach of SFLA makes it more suitable for combinatorial problems than PSO algorithm [34]. SFLA is investigated to solve different complex optimization problems, such as pick-and-place sequencing optimization [35], 0/1 knapsack problem [36], travelling salesperson problem [37], project scheduling problem [38], flow shop scheduling [39,40], optimization in cloud computing [41], optimization in social networks [42], vehicle routing problem [43] and economic load dispatch problem [44].

Several variations of SFLA are proposed in the literature to improve the efficiency of the algorithm. Centroid mutation-embedded SFLA was presented by Sharma et al. [45], where geometric centroid mutation was introduced in the algorithm to improve the convergence process. Binomial crossover-embedded SFLA [46] is presented to speed up the convergence and to improve the exploitation capabilities of the algorithm. Furthermore, an Elitism-based SFLA [47] is introduced to improve the

diversification in the algorithm. Wang et al. [48] presented a hybridized variant shuffled frog leaping differential evolution to enhance the radio time perception such that the system throughput is increased.

Edla et al. [19] presented an improved shuffled frog leaping algorithm (ISFLA) to balance the gateway load in WSNs. Authors have compared the performance of proposed ISFLA algorithm with other load balancing techniques namely node local density load balancing (NLDLB) [49], score-based load balancing (SBLB) [18], simple GA (SGA) load balancing [50] and novel GA load balancing (NGA) [51]. Authors have reported that the ISFLA is better than the mentioned state-of-art load balancing algorithms. This article identifies three research gaps as listed below:

- There is scope to improve the success of local search in the SFLA.
- Investigate the effect of probability of information exchange on the performance of SFLA.
- Incorporate problem-specific knowledge in SFLA to improve the load balancing of gateways and network lifetime.

This article presents an extension to the work presented by Edla et al. [19]. This article enhances the load balancing in WSNs with modifications in ISFLA by Edla et al. [19]. The main contributions of this article are summarized as follows:

- Like other heuristic algorithms, SFLA has components for local and global search. The local search is influenced by particle swarm optimization algorithms. The local search approach is a deterministic approach to guide the random heuristic approach [34]. The article presents three local exploration strategies to enhance the performance of SFLA.
- The proposed energy-biased load reduction phase optimizes load balancing of gateways and increases the lifetime of WSNs.
- Tested the performance of SFLA with different frog population size, probability of information exchange and the probability of energy-biased load reduction phase.
- Enhanced SFLA (ESFLA) achieves better results with respect to load balancing, network energy consumption, convergence rate, successful local explorations and first gateway die for small, medium and large datasets of WSNs.

This article is organized as follows. Related work is presented in Section 2. Section 3 is about energy models and basic SFLA. Section 4 presents the proposed ESFLA. Section 5 presents experimental details, results and discussion. Finally, Section 6 is the conclusion.

2 Related work

In the literature, different algorithms are investigated to optimize the clustering and load balancing in WSNs. LEACH is a self-organizing, distributed clustering mechanism that uses randomized rotation of CHs [52]. In this, sensor nodes use energy probability to elect themselves as CH. This randomization does not always guarantee a good cluster formation. LEACH-C presents a centralized version of LEACH [53]. In LEACH-C, CHs are selected by base station rather than self-elected CH. However, load on the base station is increased due to communication overhead of CH selection. PEGASIS presents a chaining mechanism where communication takes place only between neighbors [54]. Although this algorithm is not so stable for large networks, HEED presents an approach of selecting CHs considering residual energy of each node [55]. But in this algorithm, iterations required are very high for transmitting control packets. Intra-cluster distance minimized using particle swarm optimization based approach [56]. However, the distance to the base station was completely ignored in this approach. Guru et al. [57] have modified the PSO algorithm by presenting a boundary checking routing phase. However, the residual energy of nodes is not considered. An energy-aware PSO algorithm (PSO-C) is presented in ref. [33]. This algorithm considers various parameters, such as intra-cluster distance and the ratio of initial energy to current energy in all sensor nodes. However, the energy consumption does not take into consideration the distance from base station to CHs. Singh and Lobiyal [58] presented novel energy-aware PSO-semi distributed (PSO-SD) approach in which energy consumption for retransmission of collided packets is also considered. Hussain et al. [50] discussed genetic algorithm for cluster formation of hierarchical WSNs. In this article, the mutation operation is performed as inversion of a random single bit of chromosome. Kuila et al. [51] presented the novel genetic algorithm (NGA) to solve the load balancing problem in WSNs. NGA improves initial population generation phase where connection between sensor nodes and gateways is taken into consideration for generating chromosomes. Kuila and Jana [10] presented a novel differential evolution algorithm to increase the lifespan of first node death. Energy consumption and network lifespan of each CH are considered for cluster formation. However, the algorithm may cause energy inefficiency due to random selection of CHs. Zhang and Yang [49] presented an NLDLB algorithm that describes three steps of selecting cluster members. First, if a member node has connectivity with only one CH. Second, if the distance of the member node and CH is below range $R/2$. Third, remaining nodes are connected to CH within range with

minimum connections. Gattani and Jafri [18] presented a score-based load balancing algorithm (SBLBA) in which WSNs, lifetime is increased by controlling packet loss during transmission. In this algorithm, the node that has maximum residual energy is selected as CH. Then, it finds two best score nodes; score for each node is calculated using the ratio of residual energy and distance. Remaining nodes send the sensed data to the CH via these two best score nodes. Edla et al. [19] presented an ISFLA, which is an evolutionary approach to balance gateway load in WSNs. Also, energy-efficient fitness function is discussed where the residual energy of gateways is taken into consideration. ISFLA is compared with genetic algorithm, novel genetic algorithm, NLDLB algorithm and SBLBA. ISFLA has proved to be better. Edla et al. [32] presented a shuffled complex evolutionary approach to solve the load balancing problem in WSNs. Authors presented novel residual energy aware fitness function for solving load balancing problem. Edla et al. [32] presented improved fitness function over Edla et al. [19] and a new allocation constraint. The allocation constraint instructs to allocate a minimum number of sensors to the gateway that is farthest from base station. Table 1 presents a summary of related work.

3 Background

3.1 Network energy model

The radio signals are used in WSNs to communicate across devices. Size of the signal depends on the size of

Table 1: Literature summary of clustering algorithms for optimization problems in WSN

Objectives	Algorithms	Contributions	Results
To minimize network energy usage	LEACH	CH is selected on the basis of initial energy. It incorporates randomized rotation of CHs to share the load among the nodes	Results proved eight times better than static clustering in terms of energy consumption and first node die
To reduce the energy consumption	LEACH-C	Assignment of CHs to sensor nodes is managed by the centralized base station	Results proved that LEACH-C delivers 40% more data as compared to LEACH per unit energy
To improve the node lifetime in the WSN	PEGASIS	Presented a greedy chain-based algorithm where node transmits data only to its close node and this chain sends the data to BS	Results proved that PEGASIS improves node lifespan by 100–300%
Increase network lifetime	HEED	Presented a hybrid solution that randomly chooses CHs based on their residual energy and nodes join clusters to reduce communication costs	Results proved HEED improves network lifespan compared to LEACH
To maximize the network lifetime and data delivery at base station	PSO-C	Presented a centralized version of PSO. It focuses on reducing intra cluster Euclidean distance. Ratio of residual energy to initial energy is considered in the objective function	Results proved PSO-C improves network lifespan and data delivery compared to LEACH and LEACH-C
To maximize the network lifetime and reduce energy consumption	PSO-SD	Reduced intra cluster distance. Proposed fitness function considers the energy consumption for retransmission of collided packets	Results are proved to be better than PSO-C and LEACH-C in terms of network lifetime and average energy consumption
To perform load balancing of CHs in WSN	NGA	Presented a novel chromosome representation scheme for a genetic algorithm. Improved initial population generation by considering communication range for sensors	Results are proved to be better than simple GA, differential evolution in terms of load balancing, active sensors and energy consumption
To perform load balancing of CHs	NLDLB	Ensure that the mean square deviation value is as minimal as possible for the quantity of sensor nodes within each cluster	Novel clustering approach. Results are validated against a couple of algorithms to prove its effectiveness in balancing the load
To improve the network lifetime	SBLBA	Presented an energy aware approach to assign a score to each node. Selection of CH is carried by these scores	Controlled packet loss during transmission
To improve the network lifetime	ISFLA	Presented an evolutionary approach by improving SFLA for proper load balancing of gateways in WSNs. An energy-efficient fitness function is also proposed to achieve better solution	Results proved that ISFLA performed better in terms of load balancing, energy consumption, network lifetime as compared to GA, NGA, NLDLB, SBLBA, etc.

the data that are being transmitted. Energy is consumed in order to generate, transmit and receive these signals. To calculate the consumption of energy for these activities, an energy model from Heinzelman et al. [53] is used. Based on the distance within the transmitter and receiver, this network energy model uses an appropriate channel for the communication. Free space channel is used if the distance is below the nominal distance δ while a multipath fading channel is used if the distance exceeds the nominal distance δ. Energy consumed for transmitting the x-bit message throughout the distance d is computed as equation (1).

$$E_T(x, d) = \begin{cases} x * E_{\text{elec}} + x * E_{\text{fs}} * d^2, & \text{if } d < \delta. \\ x * E_{\text{elec}} + x * E_{\text{mp}} * d^4, & \text{if } d \geq \delta. \end{cases} \quad (1)$$

Here, E_{elec} is per bit energy utilization of the electronic circuit, E_{fs} is per bit energy utilization of the free space model and E_{mp} is per bit energy utilization of the multipath fading channel. Equation (2) demonstrates the energy consumption of a node to collect x-bit data within its range.

$$E_R(x) = x * E_{\text{elec}}. \quad (2)$$

3.2 SFLA

This subsection presents the strategy of the SFLA [19,34,59]. Figure 3 displays the flowchart of the SFLA. The steps of SFLA are described below.

1. *Initial population generation*: Set of algorithm-specific parameters namely maximum generations, memeplex count, sub-memeplex count and population size are initialized in this phase. Specified numbers of frogs are randomly generated.
2. *Fitness calculation and frog arrangements*: In this step, fitness values of all frogs are calculated with the help of fitness function, then all the frogs are arranged in ascending order according to their fitness value.
3. *Memeplex arrangements*: Sorted frogs are then distributed into a specified number of the memeplexes. Frogs are distributed like the first frog is placed in the first memeplex, the next frog is placed in the second memeplex, ith frog is placed in the ith memeplex and $(i + 1)$th frog is again placed in the first memeplex.
4. *Sub-memeplex arrangements*: Each individual memeplex is further partitioned into a specified number of sub-memeplexes.
5. *Local exploration*: Every sub-memeplex is evolved individually as explained in Figure 4. During evolution, information is exchanged between the best and the worst frog of the sub-memeplex.
6. *Convergence satisfaction test*: At the end of each iteration, convergence criteria are checked. If criteria are satisfied, further execution is stopped, otherwise, algorithms continue for the next iteration. Convergence criteria could be threshold fitness value or maximum iteration count.

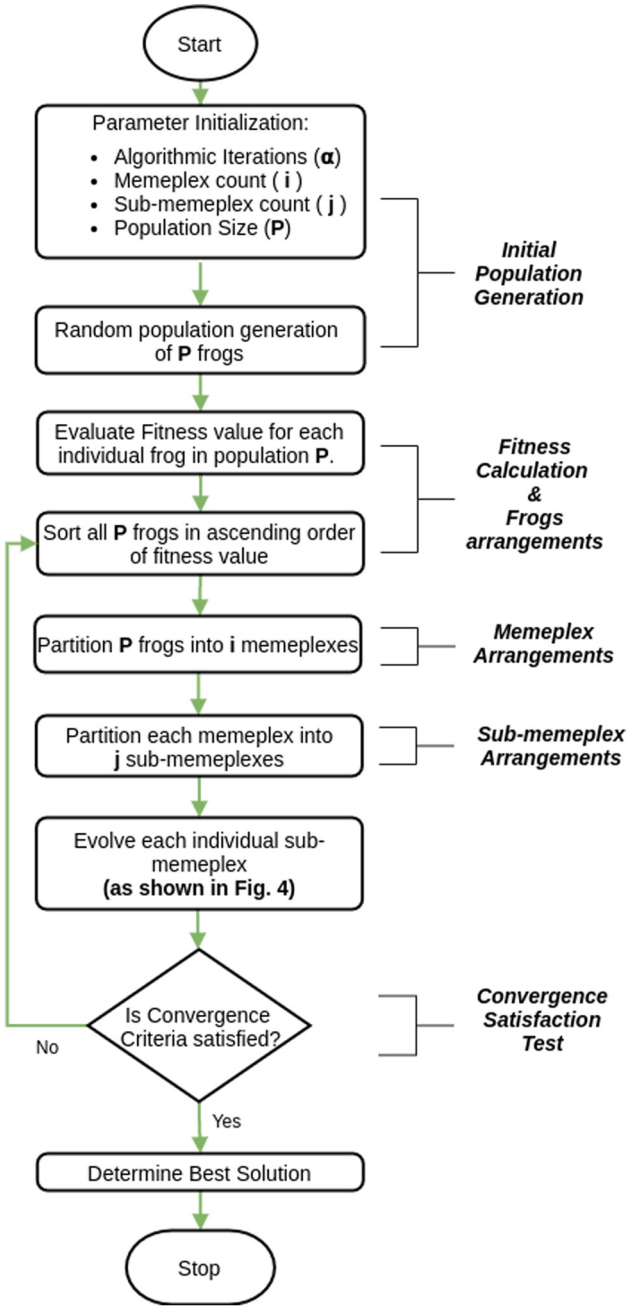

Figure 3: Flowchart of SFLA.

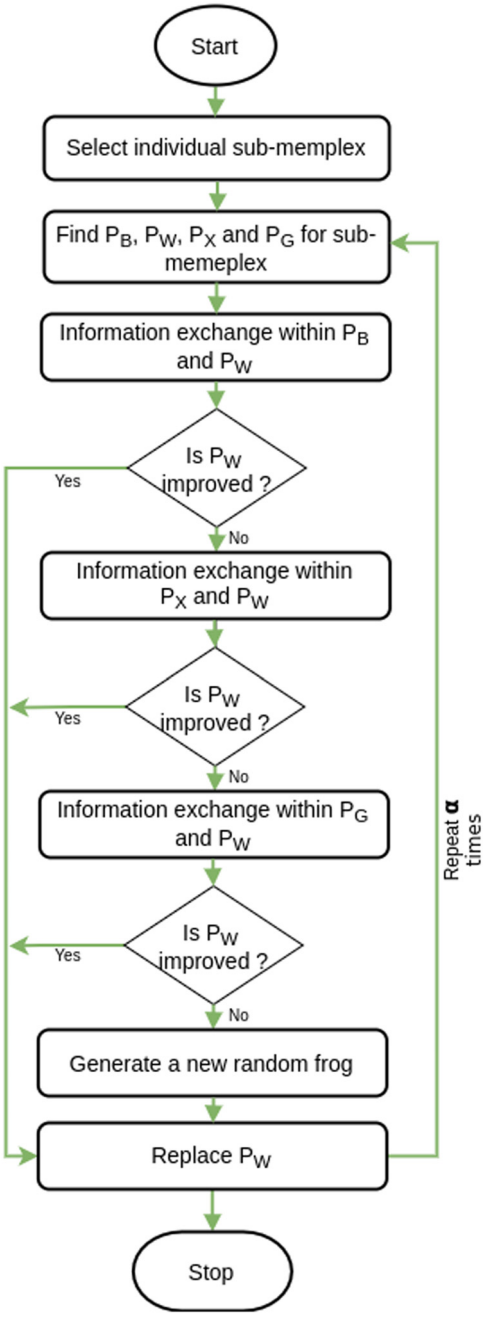

Figure 4: Local search phase in SFLA.

4 ESFLA variations to balance the gateways, load in WSNs

The evolution phenomenon of ISFLA speaks about improving the worst frog by exchanging memes with the best frog of each sub-memeplex. Since information is exchanged within best and worst frogs of each sub-memeplex, randomness in the algorithm is reduced. This restricts the local exploration capability of ISFLA. Three local exploration strategies are proposed for improving the local exploration capability. The proposed algorithm is an extension of the ISFLA presented by Edla et al. [19]. This section presents solution representation, initialization function, fitness function, steps of proposed ESFLA and the proposed energy-biased load reduction phase.

s	1	2	3	4	5	6	7	8	9	10
g	2	3	4	1	1	2	3	4	4	2

Figure 5: Sample individual frog representation.

4.1 Frog (solution) representation

Frog (solution) represents the assignment of gateways to sensors. The length of the frog is equal to the count of the sensors in WSNs. Each sensor is allocated to one of the gateways within the interaction range. A sample solution representation is shown in Figure 5. In the given representation, sensors s_4, s_5 are connected to gateway g_1 and sensors s_1, s_6, s_{10} are connected to gateway g_2.

4.2 Generation of initial populations

In the initialization phase, a specified number of frogs are generated randomly. During the generation of each individual frog, a gateway is randomly assigned to each sensor node considering the interaction range. Since gateways within interaction range are allocated to the sensor nodes, each frog generated in the initialization phase is feasible.

Example 1. Let us consider a WSN with ten sensor nodes $S = s_{10}, s_9, s_8, s_7, s_6, s_5, s_4, s_3, s_2, s_1$ and four gateways $G = g_1, g_2, g_3, g_4$. Table 2 displays the gateways and sensor

Table 2: Example of sensor nodes and gateways within its interaction range

Sensor node	Gateways within range	Sensor node	Gateways within range
s_1	g_2, g_3	s_6	g_3, g_4
s_2	g_1, g_2, g_4	s_7	g_1, g_2
s_3	g_2, g_4	s_8	g_2, g_3, g_4
s_4	g_1, g_3, g_4	s_9	g_4
s_5	g_3, g_4	s_{10}	g_1, g_4

s	1	2	3	4	5	6	7	8	9	10
g	2	1	4	3	3	4	2	3	4	1

Figure 6: A valid individual frog from the set of initial population.

nodes within the interaction range. As per Table 2, sensor node s_1 can be connected to either g_2 or g_3, sensor node s_2 can be connected to g_1, g_2 or g_4. Figure 6 presents a feasible frog for Example 1.

4.3 Fitness calculation

Fitness of each individual frog is calculated using a fitness function presented by Edla et al. [19]. Equation (3) defines the fitness function used in the proposed algorithm.

$$\text{fitness} = \left(1 - \frac{\mu_{\text{load}}}{G_{\max}}\right) + \frac{\#\,\text{heavy and under loaded gateways}}{\text{Total \# gateways}}. \tag{3}$$

Here, μ_{load} represents the mean load on all gateways. μ_{load} can be calculated using equation (4).

$$\mu_{\text{load}} = \frac{\sum_{i=1}^{i=\text{no. of gateways}} \text{Load}(G_i)}{\#\,\text{gateways}}. \tag{4}$$

For load calculation, the energy required to complete various operations by gateways is taken into consideration. Load on each gateway can be mathematically calculated with equation (5)

$$\text{Load}(G_i) = x * \frac{E_{\text{remain}}(G_i)}{E_{\text{initial}}(G_i)}, \tag{5}$$

where x is the count of bits that are to be sent by ith gateway. E_{remain} is the residual energy of gateway G_i, while E_{initial} is the initial energy of gateway G_i. E_{remain} is calculated with equation (6)

$$E_{\text{remain}} = E_{\text{initial}} - E_R(x) - E_T(x, d). \tag{6}$$

$E_T(x, d)$ and $E_R(x)$ can be calculated with equations (1) and (2), respectively.

The number of heavy and under loaded gateways is the count of gateways that are not between the maximum threshold ($T_{\max}$) and minimum threshold ($T_{\min}$). Gateways with load exceeding $T_{\max}$ are heavy loaded gateways, and gateways with a load below $T_{\min}$ are under loaded gateways. Threshold values $T_{\max}$ and $T_{\min}$ are calculated with equations (7) and (8), respectively.

$$T_{\max} = \mu_{\text{load}} + \mu_{\text{range}}. \tag{7}$$

$$T_{\min} = \mu_{\text{load}} - \mu_{\text{range}}. \tag{8}$$

μ_{range} can be calculated with equation (9) considering the difference of boundary gateway load (R) and the total count of gateways. The value of R can be calculated with equation (10). In equation (10), $G_{\max}$ is the maximum gateway load, while $G_{\min}$ is the minimum gateway load.

$$\mu_{\text{range}} = \frac{R}{\text{Total\#gateways}}. \tag{9}$$

$$R = G_{\max} - G_{\min}. \tag{10}$$

The problem is formulated as a minimization problem. Frog with minimum fitness value is better. The fitness value is based on energy consumption, so it cannot be zero or negative.

4.4 Memeplex arrangements

The number of frogs generated in the initialization phase is distributed into a specified number of memeplexes. All the frogs are arranged in increasing order of their fitness value and then they are distributed among memeplexes. The distribution of sorted frogs happens as per equation (11), where the first frog is placed in the first memeplex, ith frog is placed in the ith memeplex and $(i + 1)$th frog is again placed into the first memeplex. Assuming eight frogs generated in the initial phase are arranged in the ascending order of their fitness value and named as f1 to f8 as shown in Figure 7 and 8 shows the distribution of eight sorted frogs into two memeplexes.

$$\text{memeplex}(f) = \text{pos}(f) \bmod m. \tag{11}$$

4.5 Sub-memeplex arrangements

Each memeplex is further broken down into j sub-memeplexes. Value for j is randomly selected from factors of memeplex size. Memeplex size is the count of frogs present in a memeplex. In Figure 8, memeplex size is 4. Factorials of 4 are 1, 2 and 4. So, the number of sub-memeplexes can be chosen from number 1, 2 or 4. Figure 9 shows partition of each memeplex from Figure 8 into two sub-memeplexes. The total number of sub-memeplexes is four, i.e., (i_1, j_1), (i_1, j_2), (i_2, j_1), (i_2, j_2).

1	2	3	4	5	6	7	8	9	10
3	2	4	1	3	4	2	2	4	1

(f1)

1	2	3	4	5	6	7	8	9	10
2	4	2	3	4	3	1	4	4	1

(f2)

1	2	3	4	5	6	7	8	9	10
2	4	2	1	3	4	1	3	4	4

(f3)

1	2	3	4	5	6	7	8	9	10
3	1	4	3	4	3	2	4	4	1

(f4)

1	2	3	4	5	6	7	8	9	10
2	4	2	3	4	3	1	2	4	1

(f5)

1	2	3	4	5	6	7	8	9	10
3	2	4	1	3	4	2	3	4	4

(f6)

1	2	3	4	5	6	7	8	9	10
3	1	4	3	4	4	2	2	4	1

(f7)

1	2	3	4	5	6	7	8	9	10
2	2	4	1	3	3	1	4	4	1

(f8)

Figure 7: Sample feasible frogs (solutions) generated in initialization step.

Memeplex 1	f1	f3	f5	f7
Memeplex 2	f2	f4	f6	f8

Figure 8: Example of partitioning step (partitioning of frogs into memeplexes).

	Submemeplex (N1)		Submemeplex (N2)	
Memeplex (M1)	f1	f3	f5	f7
Memeplex (M2)	f2	f4	f6	f8

Figure 9: Example of partitioning of two memeplex into two sub-memeplexes.

4.6 Local exploration phase

This stage performs evolution of each sub-memeplex with information exchange between selected frogs. In ISFLA, the evolution of each sub-memeplex is performed with information exchange from best to the worst frog. We present three approaches to choose source and target frogs to contribute to the sub-memeplex evolution phase. Instead of selecting the best and worst frogs for information exchange, the presented approaches focus on giving scope to other frogs to take part in the exchange process. After information exchange, a novel energy-biased load reduction phase is introduced to increase the lifetime of gateways in WSNs. In this phase, residual energy of gateways is taken into consideration and a necessary biased decision is made to save the dying gateway.

4.6.1 ESFLA with global guided

This modified version of ISFLA is termed as Enhanced Shuffled Frog Leaping Algorithm-Global Guided (ESFLA-GG). Figure 10 shows a flowchart of this approach. In ESFLA-GG approach, the worst frog of the sub-memeplex is replaced by the global best frog. Replacing the worst frog with the global best frog in all the sub-memeplexes does not provide randomness. To maintain a diversity of frogs, this replacement is carried out only in half of trailing sub-memeplexes before the evolution process is started. We denote *TS* as a set that contains only half of the trailing sub-memeplexes as expressed in equation (12). After replacement, the new best frog in the sub-memeplex is appointed as a source frog and the worst frog is appointed as a target frog for information exchange.

$$TS = \{s | s \text{ belongs to trailing half sub-memeplexes}\}, \quad (12)$$

where $s \rightarrow$ individual sub-memeplex.

4.6.2 ESFLA with stochastic universal sampling (ESFLA-SS)

Baker [60] presented a zero-bias sampling algorithm termed as stochastic universal sampling (SUS). Basically,

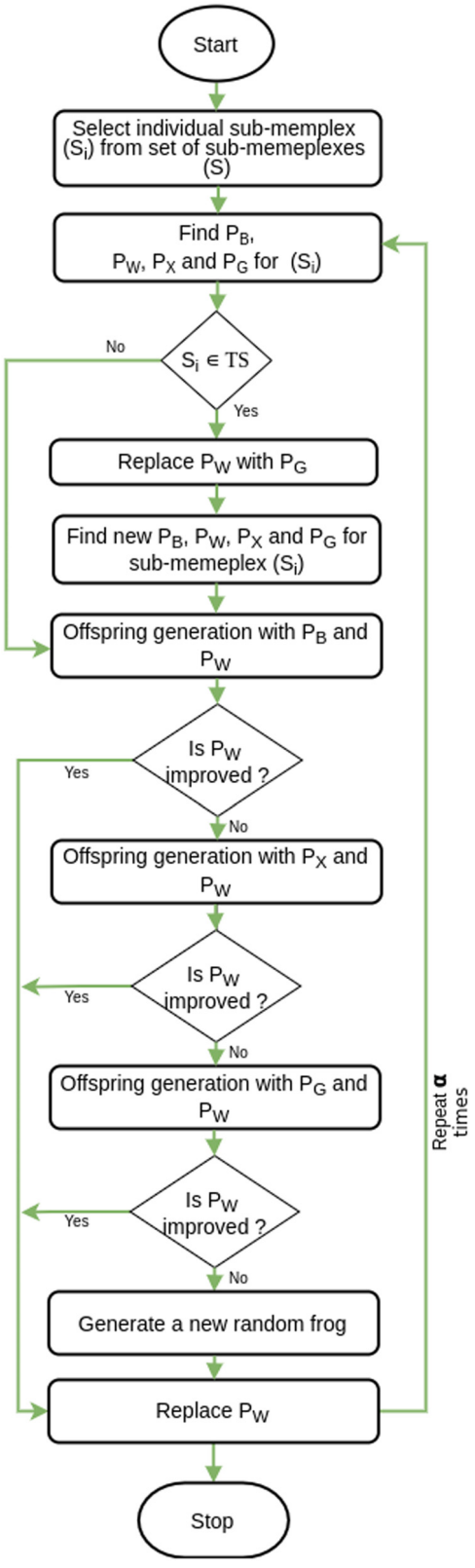

Figure 10: ESFLA with global guided.

SUS is employed to select N equally spaced samples from a given population. All the individuals in the population are mapped on a line segment ranging from 0 to 1, such that the area occupied by the individual is equal to its fitness proportion (in ratio to sum of all individual's fitness). Greater the fitness more the area allocated to the individual on the segment. To place the N pointers, a random value r is generated within range $(0, 1/N)$, the first pointer is placed on the generated value r, the remaining pointers are placed with a space of $1/N$.

Algorithm 1: Generating fitness probability scale

```
Input: Array of solutions with its Fitness values Fitness[]
Output: Fitness Probability Scale within range 0 to 1
Procedure getFitnessProbabilityScale(Fitness[]):
    N := number of frogs := len(Fitness);
    /* Use exponential fitness for minimization problem */
    for index := 1 to N do
        ExpFitness[index]:=exponential(−1 * Fitness[index])
    end
    F:=sum of ExpFitness
    Scale:=0
    for index := 1 to N do
        Scale:=Scale + ExpFitness(index)/F
        ScaleFitness[index] := Scale
    end
    ScaleFitness ← Sorted (ScaleFitness)
    return ScaleFitness
```

Algorithm 2: Choose the source and target frog by stochastic universal sampling

```
Input: Population P, Fitness of Population Fitness[], Number of Frogs to be selected S.
Output: Selected Source and Target Frogs
Procedure getStochasticFrogs(P, Fitness[], S):
    S:=Number of samples to be selected
    //Number of samples are 2 in this case
    N := number of frogs:=len(P)
    Scale := getFitnessProbabilityScale(Fitness)
    R1:=random(0, 1/S)
    Pointer to source frog
    R2 := R1 + 1/S
    Pointer to target frog
    Frogs[ ]←Initialize Empty Array
    for i := 1 to N do
        if R1 < = Scale(i) then
        | Frogs['sourceFrog'] := P[i]
        end
        if R2 < = Scale(i) then
        | Frogs['targetFrog'] := P[i]
        end
    end
    return Frogs
```

We employed the SUS technique to select the source and target frog from the sub-memeplex, i.e., $N = 2$. The frogs are sorted in increasing order of their fitness values and are mapped to a line segment of range (0, 1). Since $N = 2$, a random value r is generated within (0, 0.5). Frog at r is selected as source frog and frog at $r + 0.5$ is selected as target frog. The selected target frog will obviously be weaker than the source frog. Figure 11 shows the flowchart of this approach.

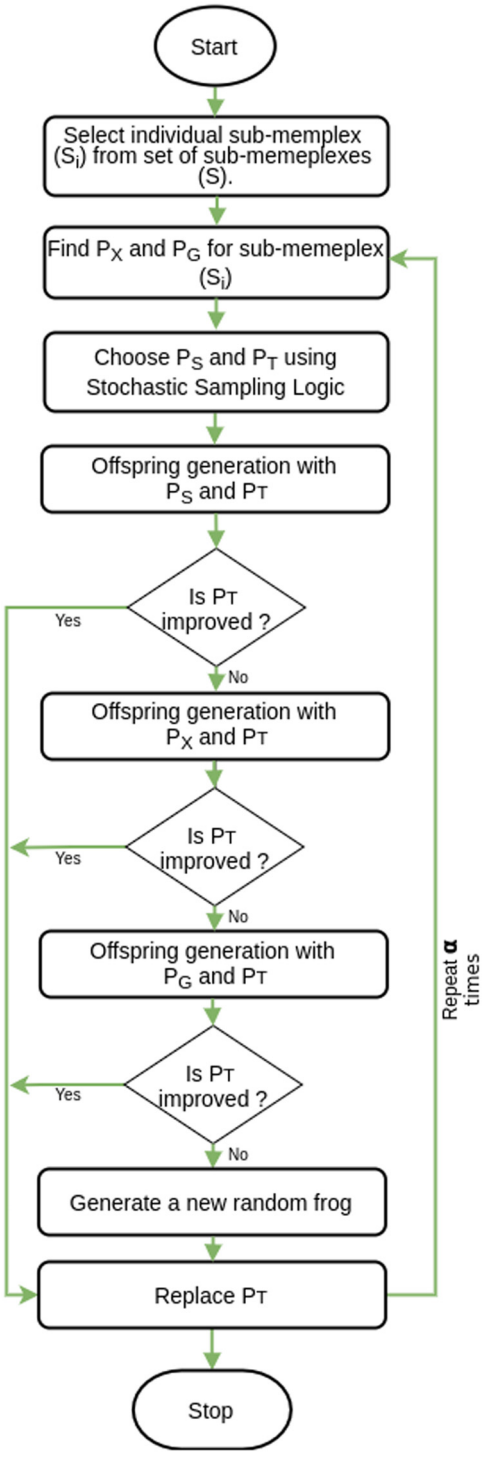

Figure 11: ESFLA with SUS.

4.6.3 ESFLA with roulette wheel (RW) selection (ESFLA-RW)

Pencheva et al. [61] used an RW-based selection approach for a genetic algorithm. In the RW approach, the circular wheel is partitioned among probabilities of all solutions just like a line segment in SUS. Contrary to SUS, RW could choose only one item at a time. A pointer is pointing toward the fixed position when the wheel is rotated. The solution is selected whose region is pointed by a pointer after the wheel is stopped. In RW, fitness values of all the solutions are converted into roulette probabilities. Better fitness frog is more likely to get selected. In this approach, a probabilistic RW is created for each sub-memeplex. Each frog in the sub-memeplex is assigned with probability values within the range 0 to 1. Better the fitness of the frog, the more the probability range allocated to the frog. This increases the probability of a better frog getting selected for information exchange. One frog from the sub-memeplex is selected using an RW rotation. This selected frog is considered as a source of information exchange. The worst frog from the sub-memeplex is considered as the target of information exchange. Figure 12 shows the flowchart of this approach.

Algorithm 3: Choose the source frog by RW selection

Input: Population *P*, Fitness of Population *Fitness*[], Target frog *Tf*.
Output: Selected Source Frog for information exchange

```
1  Procedure getRouletteFrog(P, Fitness[], Tf ):
2      N := number of frogs:=len(P)
3      Scale:=getFitnessProbabilityScale(Fitness)
       /* Generate Random number R such that it does not
4      returns already select target frog Tf        */
5      R := random(0, 1)
6      Frogs[ ]←Initialize Empty Array
7      for i := 1 to N do
8          if R < = Scale(i)then
9          | sourceFrog := P[i]
10         end
       end
11     return sourceFrog
```

Once the source and target frogs are selected from any of the above specified local exploration approaches, the information exchange is performed and new offspring

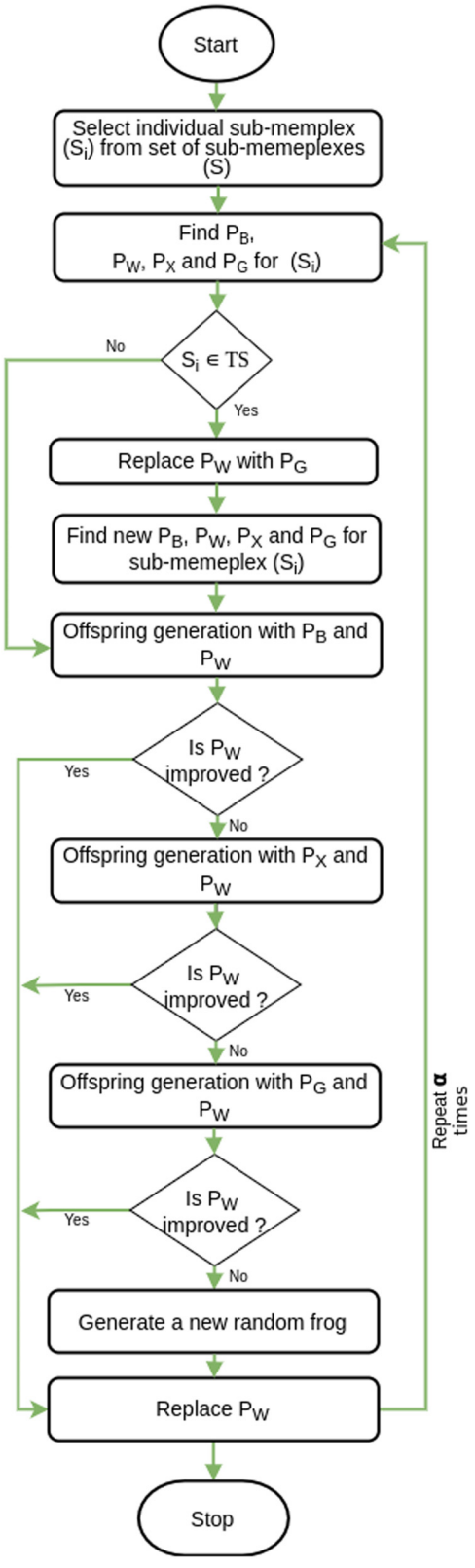

Figure 12: ESFLA with roulette wheel (ESFLA-RW) selection.

is generated to replace the existing target frog. The information exchange is a single point crossover operator. For information exchange, a random point p is selected such

1	2	3	4	5	6	7	8	9	10
2	2	4	1	3	4	2	3	4	4

Figure 13: Offspring generated using roulette wheel (RW) selection.

that $1 < p < n$, in which n is the count of sensor nodes. All the gateway bits from p to n of source frog are copied to the target frog. Each sub-memeplex will go through the same process. Taking as an example, let us consider we get (f6) as source frog in local exploration and (f8) as target frog. Figure 13 shows newly generated offspring assuming random point $p = 4$ for single point crossover operation.

To improve the solution quality, we proposed a novel energy-biased load reduction phase for gateway. In this phase, the residual energy of gateways is taken into consideration in order to increase the lifetime of gateways in WSNs. The energy consumption of gateways is estimated using equation (6) for the newly generated offspring in the above phase. Residual energy of gateways is initial energy minus estimated energy consumption. The gateway with lowest estimated residual energy is selected to reduce the load so that its lifetime is increased.

Algorithm 4: Proposed energy-biased load reduction phase

Input: Solution S, Gateways available energies GE, Gateways in Range of sensors R
Output: Improved Solution
Procedure ProposedLoadReduction(S, GE):
  energyConsumed ← Energy consumed by gateways for solution S
  foreach *gateway in Gateways* **do**
    GE[gateway] = GE[gateway] – energyConsumed[gateway]
  end
  G_{min} := min(gatewayEnergies)
  //Find gateway with minimum residual energy
  /* Find farthest distance sensor S_f that is allocated to G_{min} but can be allocated to another gateway */
  S_f := getFarthestSensor(G_{min})
  choose randomly $G_{new} \in R(S_f)$ such that G_{new} != G_{min}
  $S[S_f] := G_{new}$
  //Allocate new gateway to the farthest sensor
  return S

To reduce the load of the dying gateway, the farthest sensor node from this selected gateway in terms of Euclidean distance is deallocated. The deallocation reduces the energy consumption of this gateway. This deallocated

1	2	3	4	5	6	7	8	9	10
2	2	4	1	3	4	2	3	4	4

Before

1	2	3	4	5	6	7	8	9	10
2	2	4	1	3	4	1	3	4	4

After

Figure 14: Solution modified after energy-biased load reduction phase.

sensor is allocated to another gateway that lies within its range. As the allocation of gateway is performed considering the range constraint, it can be stated that the offspring generated in the proposed energy-biased load reduction phase is a feasible solution. Carrying forward the sample offspring in Figure 13, let us assume that the gateway g_2 has the lowest residual energy and sensor node s_7 is farthest in terms of Euclidean distance from g_2. To reduce the load from g_2, sensor node s_7 is deallocated from gateway g_2 and s_7 is allocated to another gateway within its range. Taking reference to Table 2, s_7 is allocated to gateway g_2. Figure 14 shows the effect of these operations.

5 Experimental details, results and discussion

5.1 Simulation details

A program is created to generate input datasets randomly. Number of sensor nodes, number of gateways, distribution area size and threshold communication range of sensors with gateways are passed as input to this program. The data generator creates input by random distribution of a specified number of sensors and gateways within a given area. Total 15 input datasets are created using the dataset generator program. Generated input datasets are categorized into three categories, namely small, medium and large, based on the count of sensors and gateways. A set of 15 inputs are shown in Table 3. Input 1–5 belongs to a small dataset category, input 6–10 belongs to a medium dataset and input 11–15 belongs to a large dataset. Input datasets have various parameters. Maximum threshold distance (m) is the maximum distance up to which sensors can transmit data, area of deployment (m × m) is the spread of a network in a geographical area, base station location is the x and y coordinates of the base station in geographical location with reference to area of deployment. Number of sensor nodes and number of gateways available in the network are given to the algorithm with their respective coordinates.

For experimentation, each dataset is considered with equal and unequal load on sensors. Equal load means each sensor node transmits the same amount of data

Table 3: Experimental scenario

Input number	Max threshold distance (m)	Area of deployment (m × m)	Base station location (x, y)	Count of sensor nodes	Count of gateways
1	10	50 × 50	(48, 25)	50	10
2	10	50 × 50	(48, 25)	100	20
3	10	50 × 50	(48, 25)	200	30
4	10	100 × 100	(96, 50)	500	50
5	10	100 × 100	(96, 50)	750	65
6	10	100 × 100	(96, 50)	1000	80
7	15	200 × 200	(196, 100)	1200	100
8	15	200 × 200	(196, 100)	1400	120
9	15	200 × 200	(196, 100)	1600	140
10	15	200 × 200	(196, 100)	1800	160
11	15	300 × 300	(296, 150)	2000	150
12	15	300 × 300	(296, 150)	2250	175
13	15	300 × 300	(296, 150)	2500	200
14	15	300 × 300	(296, 150)	2750	225
15	15	300 × 300	(296, 150)	3000	250

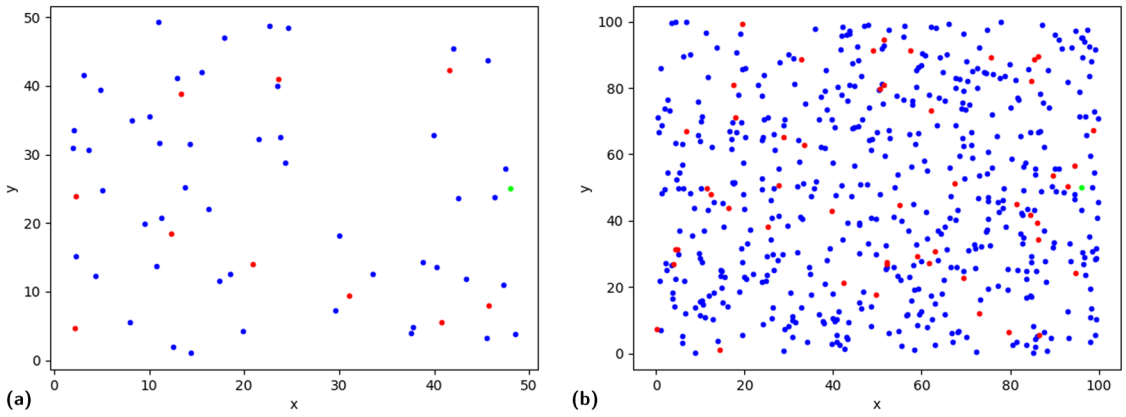

Figure 15: Deployment of sensor nodes and gateways. (a) Deployment of 50 sensor nodes in 50 × 50 m^2 area. (b) Deployment of 500 sensor nodes in 100 × 100 m^2 area.

bits while unequal load means that sensor nodes transmit data of varying length. Unequal load is chosen randomly for each sensor node from a given set of load sizes. Two sample inputs generated with the program are shown in Figure 15. Red dots indicate gateways, green dots indicate sensor nodes and a blue dot indicates base station.

Table 4 presents simulation parameters used in experiments. Default $E_{initial}$ is the initial energy in the gateways at the start of the first round; however, this value changes after the end of each round depending on gateway's energy consumption. The sensor data length is the number of bits transmitted by sensor nodes. The number of memeplexes and sub-memeplexes are chosen randomly as explained in Sections 4.4 and 4.5, respectively.

Table 4: Simulation parameters

Initial gateway energy ($E_{initial}$)	10,000,000 nJ
Data transfer length (equal load)	8 bits
Sensor data length (unequal load)	rand(4, 6, 8, 10, 12, 14) bits
Energy required by circuit (E_{elec})	50 nJ/bit
Energy required by free space (E_{fs})	10 pJ/bit/m^2
Energy required by MP (E_{mp})	0.001 pJ/bit/m^4
Frog population size (PopSize)	100
Memeplex count	rand(factorialSet (PopSize))
Sub-memeplex count	rand(factorialSet(PopSize/ MemeplexCount))
Max convergence iterations (α)	50
Max roulette iterations (maxRouletteIterations)	5

5.2 Results of ESFLA variations

Experiments are conducted for all the 15 input datasets described in Section 5.1. The results of the proposed ESFLA are compared with ISFLA presented by Edla et al. [19]. SFLA is an evolutionary algorithm. Evolutionary algorithms are probabilistic in nature. Traditional algorithmic analysis is not applicable to these algorithms. The measure performance of evolutionary algorithms best value, mean value and standard deviation is used. Suganthan et al. [62] presented six metrics for evaluating the performance of these algorithms for mathematical benchmark function. These six parameters are namely convergence graph, average number of function evaluations (NFEs), success rate, acceleration rate and improvement. This article presents the SFLA for application in WSN. Following metrics are used for performance analysis. Algorithmic metrics used are successful local explorations, best fitness values, average fitness values and convergence graph. Application-specific metrics used are network energy consumption, load balancing and first gateway die.

5.2.1 Average best fitness value

Load balancing is derived from the fitness value of the best solution obtained at the last iteration. Better fitness value indicates better load balancing. The best fitness value obtained from the algorithm at last iteration is measured. For each dataset, the program is executed ten times and an average of all ten best results are taken into consideration to neutralize the randomness effect. Figure 16a and b shows comparison of proposed ESFLA-GG approach with ISFLA for equal and unequal load,

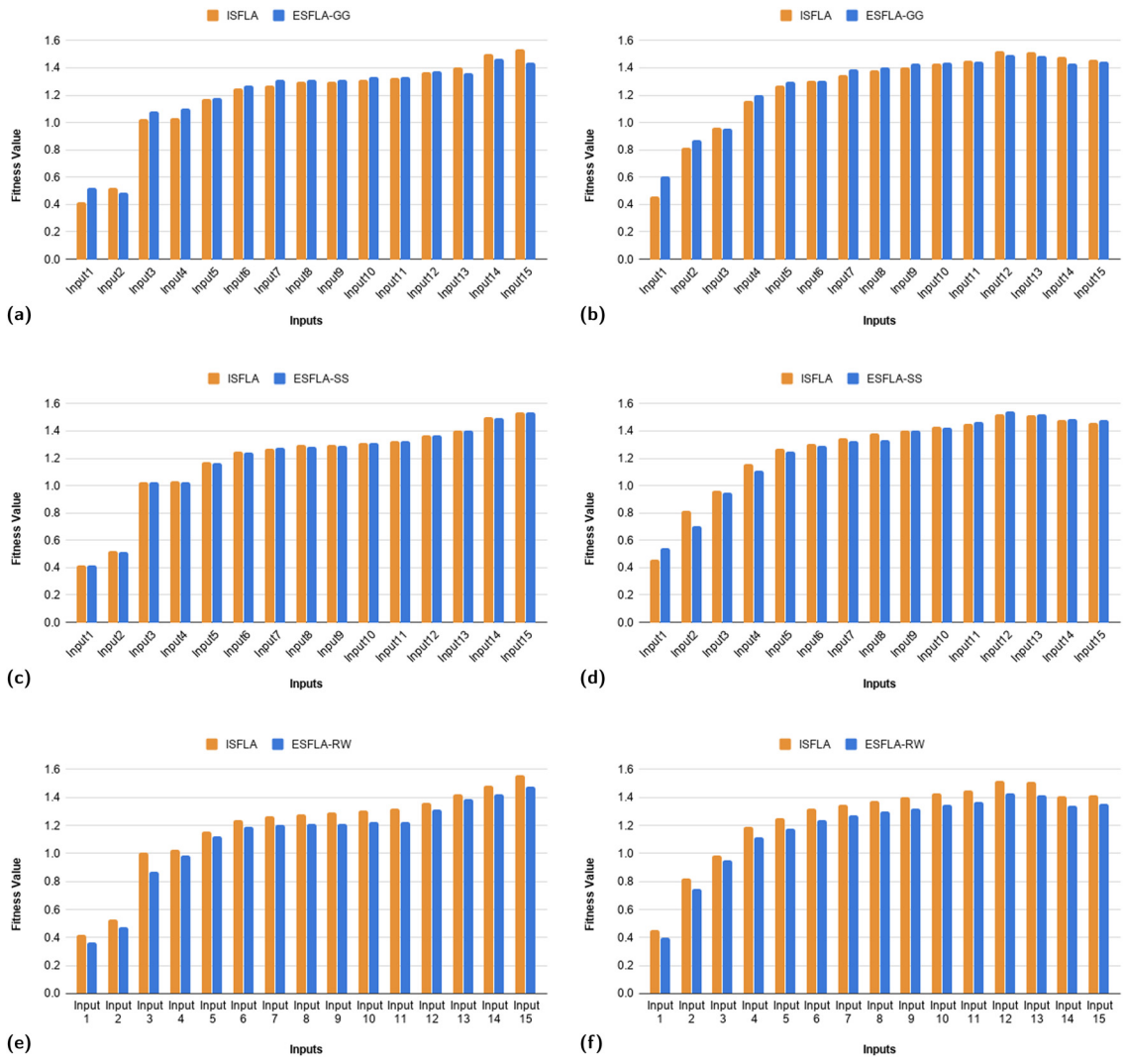

Figure 16: Comparison of proposed ESFLAs using average best fitness value. (a) ESFLA-GG equal load, (b) ESFLA-GG unequal load, (c) ESFLA-SS equal load, (d) ESFLA-SS unequal load, (e) ESFLA-RW equal load and (f) ESFLA-RW unequal load.

respectively. For a few datasets, ESFLA-GG is found slightly better than ISFLA. Results show that ESFLA-GG does not guarantee performance improvement over ISFLA for all cases. ESFLA-GG presents a deterministic approach by replacing the worst frog with the global best frog in half of the trailing sub-memeplexes. Results show that the proposed deterministic approach does not guarantee improvement in local exploration every time due to reduction in population diversity.

Figure 16c and d shows comparison of proposed ESFLA-SS approach with ISFLA for equal and unequal load, respectively. For a few datasets, ESFLA-SS is found slightly better than ISFLA. Results from Figure 16 show that ESFLA-SS provides slight improvement over ISFLA for the majority of the datasets. Figure 16e and f shows comparison of proposed ESFLA-RW approach with ISFLA for equal and unequal load, respectively. ESLFA-RW is found better for all small, medium and large datasets with equal and unequal load. The performance improvement of ESFLA-RW is significant over ISFLA since in ESFLA-RW the improvement is always done in the worst frog contrary to ESFLA-SS where the target frog is randomly selected for improvement. In the literature, it is reported that the performance of many heuristic algorithms is subjected to problem instances. Change in the problem instance size and complexity affected the algorithm performance. The proposed local exploration strategies of ISFLA are not instance specific. The ESFLA-RW shows significant performance improvement for small, medium and large instances with equal and unequal load.

5.2.2 Success of local exploration

To verify the improvement of proposed ESFLA approaches over ISFLA, we calculated the percentage of successful local exploration for all sub-memeplexes. A local exploration is said to be successful if a better frog is obtained by information exchange within the sub-memeplex itself. Figure 17 shows the results for local exploration performance of ISFLA and ESFLA for equal and unequal load for input 15. Results show that the proposed three local exploration strategies are better than ISFLA. The successful local exploration percentage of ISFLA and ESFLA variations increases with an increase in the number of frogs. It is observed that ESFLA-RW has improved its local exploration capability as compared to all other algorithms. This improvement is due to the increased exposure to choose frog within the sub-memeplex rather than always exchanging information from best frog to worst frog.

Successful local exploration demonstrates how proposed variations improved the local exploration capability of the algorithm, which means how many times the information exchange within sub memeplex itself has improved the fitness of target frog. In proposed ESFLA-RW, the information is exchanged within the best and random frog selected from the RW, which improved the exploration capability of the algorithm and led to achieve better results. Figure 18 shows the performance of ISFLA, ESFLA-GG, ESFLA-SS and ESFLA-RW (from left to right) for input 15. The graphs in Figure 18a and b show the distribution of fitness values of the entire population. The boxplot representation of the population shows minimum, maximum, median, first quartile and third quartile. Results indicate that the proposed ESFLA is better than ISFLA with respect to minimum and median values. Population diversity is greatly reduced in ESFLA-GG due to partial exchange of the worst frogs with the global best frog. ESFLA-SS shows the highest population diversity compared to all other approaches since high randomness is introduced in selecting source and target frogs. ESFLA-RW provides the best fit frog (minimum value) over all other approaches. ESFLA-RW shows improvement in overall population compared to ISFLA.

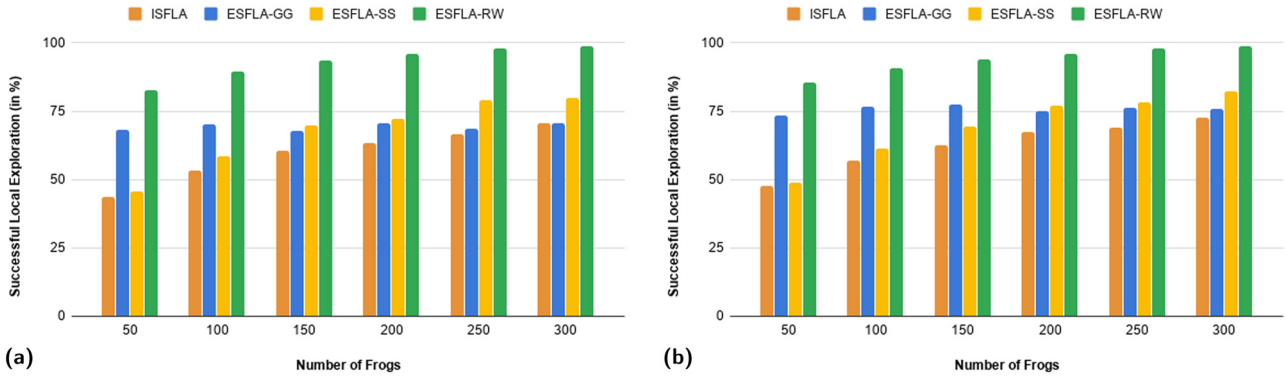

Figure 17: Comparison of proposed ESFLAs using success of local exploration: (a) equal load and (b) unequal load.

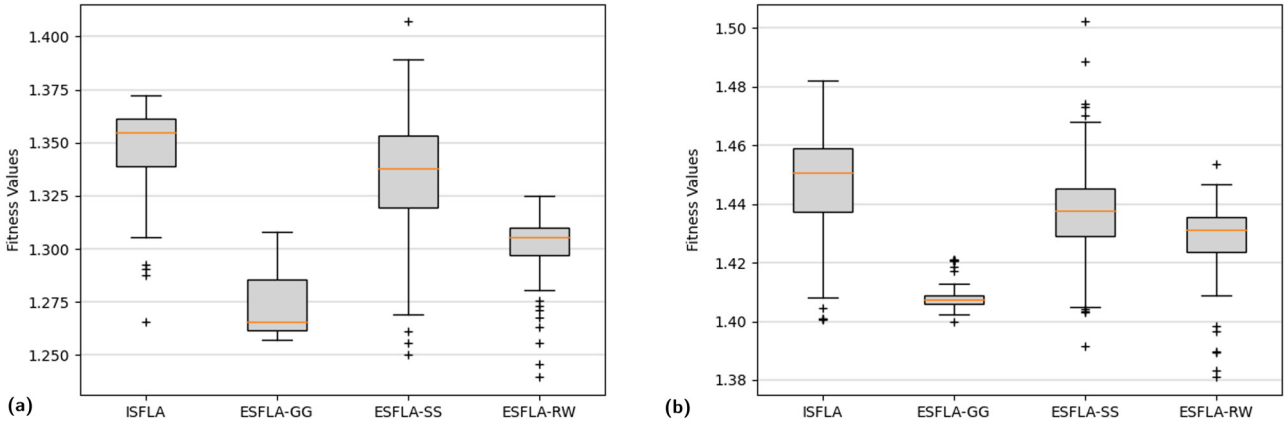

Figure 18: Performance analysis of proposed ESFLAs. (a) Equal load, (b) unequal load.

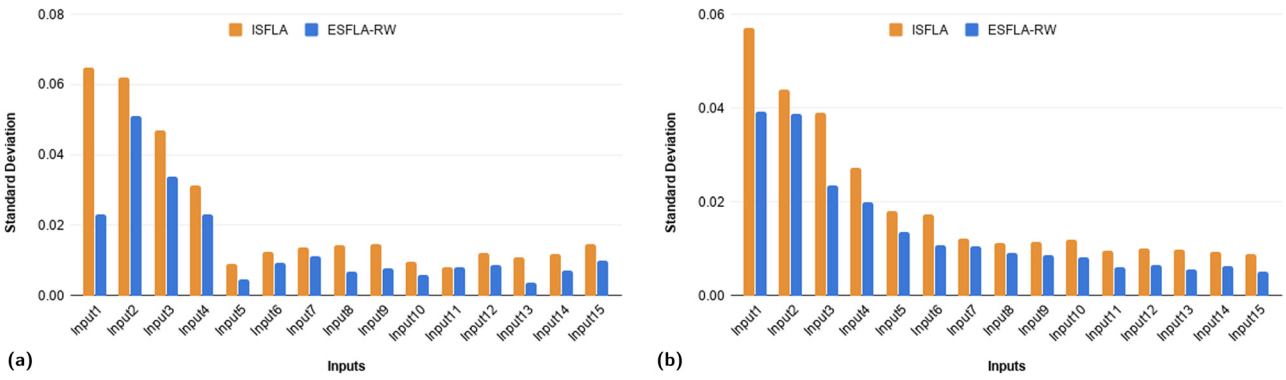

Figure 19: Comparison of proposed enhanced SFLAs using standard deviation. (a) Equal load, (b) unequal load.

ESFLA-RW performed better than ESFLA-GG and ESFLA-SS. In the next subsections, comparisons are presented in ISFLA and ESFLA-RW. Figure 19 shows the standard deviation of population fitness at the last iteration. It is observed that standard deviation of ESFLA-RW is better than ISFLA. It indicates the progress of the entire population towards the best region.

Population diversity has a strong impact on performance of population-based nature-inspired/bio-inspired algorithms. Good population diversity is necessary to explore the search space. In the literature, it is reported that poor population diversity leads to premature convergence of many evolutionary algorithms such as genetic algorithms. Due to premature convergence, the algorithm faces difficulties or fails to reach a global optimal solution. SFLA forced the information exchange between best frog and worst frog. The proposed local exploration strategies changed this restrictive selective process of frog selection. Results show that the proposed methodology improved the performance of SFLA.

5.2.3 Execution time

We compared the execution time of the proposed ESFLA-RW with ISFLA. The obtained results are shown in Figure 20 for small, medium and large datasets with equal load. Results show that the proposed ESFLA-RW takes slightly more time than ISFLA. This lag is due to the introduction of additional RW selection phase. Similar results are obtained for unequal load datasets.

5.2.4 Energy consumption

Total energy consumed by WSNs is the sum of energy consumed by gateways and sensors. The energy is required to send and receive data among sensors, gateways and base stations. Energy consumption (in Nanojoules) is calculated using the best solution generated by algorithms. Table 5 shows the results for energy consumption of ESFLA-RW and ISFLA. Energy consumption specifies what amount of energy is consumed by the network when the sensor to gateway assignments is made on the basis of the best solution obtained by these algorithms. We can see that the best solution obtained from ESFLA-RW consumes less energy as compared to ISFLA. The proposed ESFLA-RW consumes less energy than the ISFLA.

5.2.5 Results of proposed energy-biased load reduction phase

To demonstrate the effect of the energy-biased load reduction, we introduced the proposed load reduction phase into ISFLA and termed it as ISFLA – enhanced load reduction (ISFLA ELR). Nodes and gateways in WSN operates in battery power. The network lifetime depends on the battery. In the literature, different methods are investigated to increase the network lifetime. This article presents load balancing of gateways to minimize the energy requirements in WSN. Network life of WSN is defined as the time until the first gateway dies. To verify the improvement in gateway lifetime, we recorded the round number where the first gateway dies. Gateway is said to be dead when there is no residual energy in gateway. We have compared the results among ISFLA, ISFLA-ELR and ESFLA-RW. Experiments are performed for one dataset from each category of small, medium and large datasets. Input 1, from small dataset, input 6 from medium dataset and input 11 from large dataset are used. A violin plot representation is used to show the residual energies of gateways before and after the load reduction phase in Figure 21 for input 15. A violin plot is similar to boxplot but gives a better idea

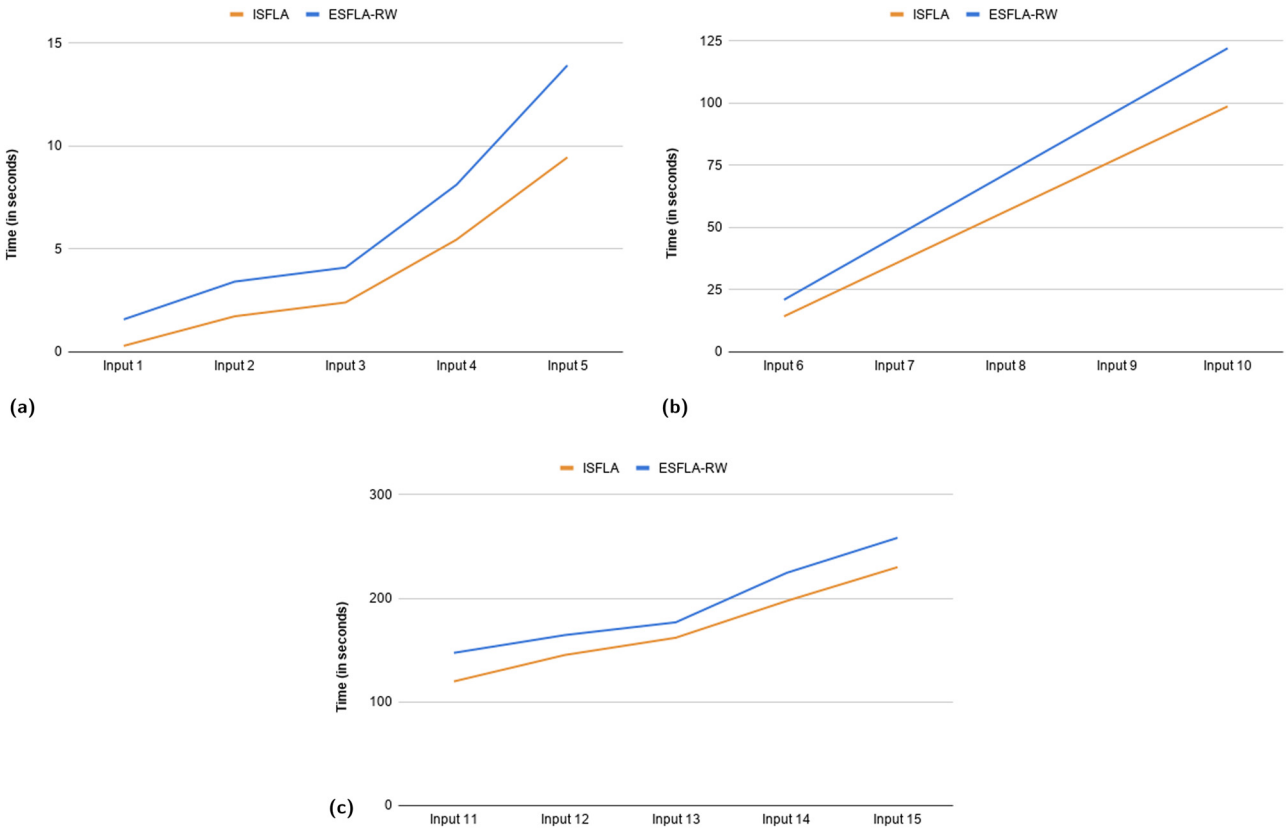

Figure 20: Comparison of ISFLA and proposed ESFLA-RW using execution time. (a) Small dataset, (b) medium dataset, (c) large dataset.

about the state of population. The minimum value in the graph shows the lowest residual energy in the gateway. It can be observed that the minimum gateway energy is increased after applying load reduction phase leading to increase in network lifetime. Table 6 shows the obtained results. It is observed that the lifetime of a gateway is increased from 9 to 15%.

Table 5: Comparison of ISFLA and ESFLA-RW using energy consumption (in Nanojoules)

	Equal load		Unequal load	
Inputs	**ISFLA**	**ESFLA-RW**	**ISFLA**	**ESFLA-RW**
Input 1	60,707	60,595	70,128	64,056
Input 2	121,178	121,135	135,710	131,736
Input 3	242,635	242,474	271,522	262,315
Input 4	685,200	683,397	786,166	751,549
Input 5	989,371	986,837	1,127,377	1,064,412
Input 6	1,360,238	1,357,708	1,543,514	1,530,097
Input 7	7,442,817	7,405,117	8,417,233	8,283,581
Input 8	13,524,798	13,452,526	15,290,417	15,037,068
Input 9	19,606,859	19,499,935	22,163,759	21,790,554
Input 10	25,688,760	25,547,345	29,037,112	28,544,040
Input 11	31,770,313	31,594,754	35,909,979	35,297,526
Input 12	34,615,050	34,285,233	39,426,381	38,836,686
Input 13	39,115,236	38,751,823	43,399,021	42,878,173
Input 14	41,688,951	41,410,116	47,550,268	46,645,722
Input 15	46,493,802	46,187,884	52,914,409	52,009,816

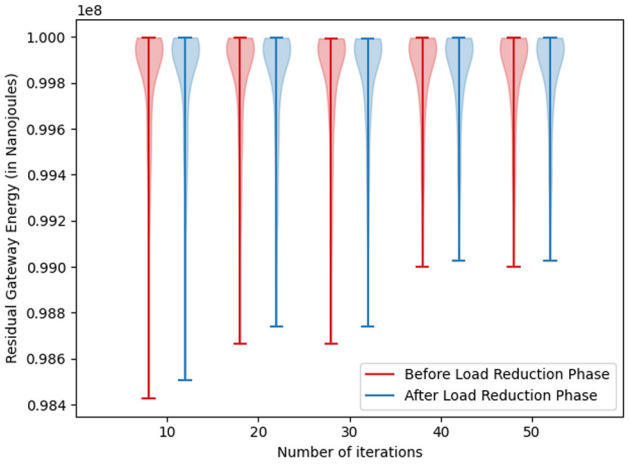

Figure 21: Residual energies of gateways before and after the load reduction phase.

Table 6: Comparison of ISFLA, ISFLA-ELR and ESFLA-RW using first gateway die measure

Inputs	ISFLA	ISFLA-ELR	ESFLA-RW	% improvement in ISFLA-ELR	% improvement in ESFLA-RW
Input 1	21,420	23,268	23,331	9	9
Input 6	4802	5436	5498	13	14
Input 11	136	157	155	15	14

5.2.6 Convergence of algorithms

Convergence demonstrates how rapidly fitness is improved with respect to algorithm iterations. Experiments are performed for two datasets from each category of small, medium and large datasets. Input 1 and input 5 from a small dataset, input 6 and input 10 from medium dataset, while input 11 and input 15 from large dataset are used. The best fitness value obtained at each iteration is measured. Figure 22 shows that the convergence of proposed ESFLA-RW is better than ISFLA. Average NFEs and number of iterations required for convergence are proportional. Enhanced versions of SFLA required less NFEs.

5.3 Effect of algorithm-specific parameters on ISFLA and ESFLA-RW

5.3.1 Effect of frog population size

Experiments are performed for three input datasets, one from each category of small, medium and large. The best fitness value obtained from ESFLA-RW by varying the frog population sizes is measured with equal and unequal load on sensors. Results from Figure 23 indicates that better fitness value is obtained with increase in the number of frogs.

Experiments are conducted to test the effect of the frog population size on convergence performance of

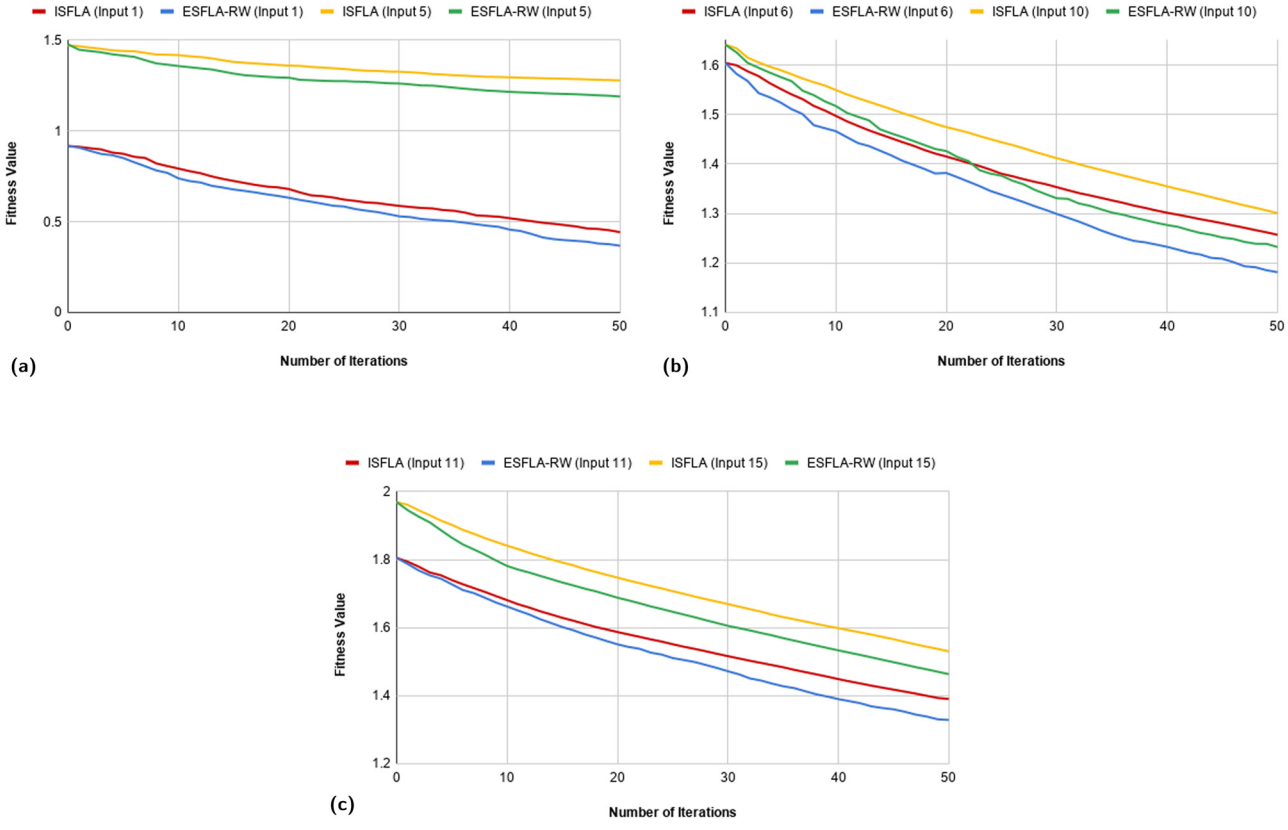

Figure 22: Convergence of ISFLA and proposed ESFLA-RW. (a) Small dataset, (b) medium dataset, (c) large dataset.

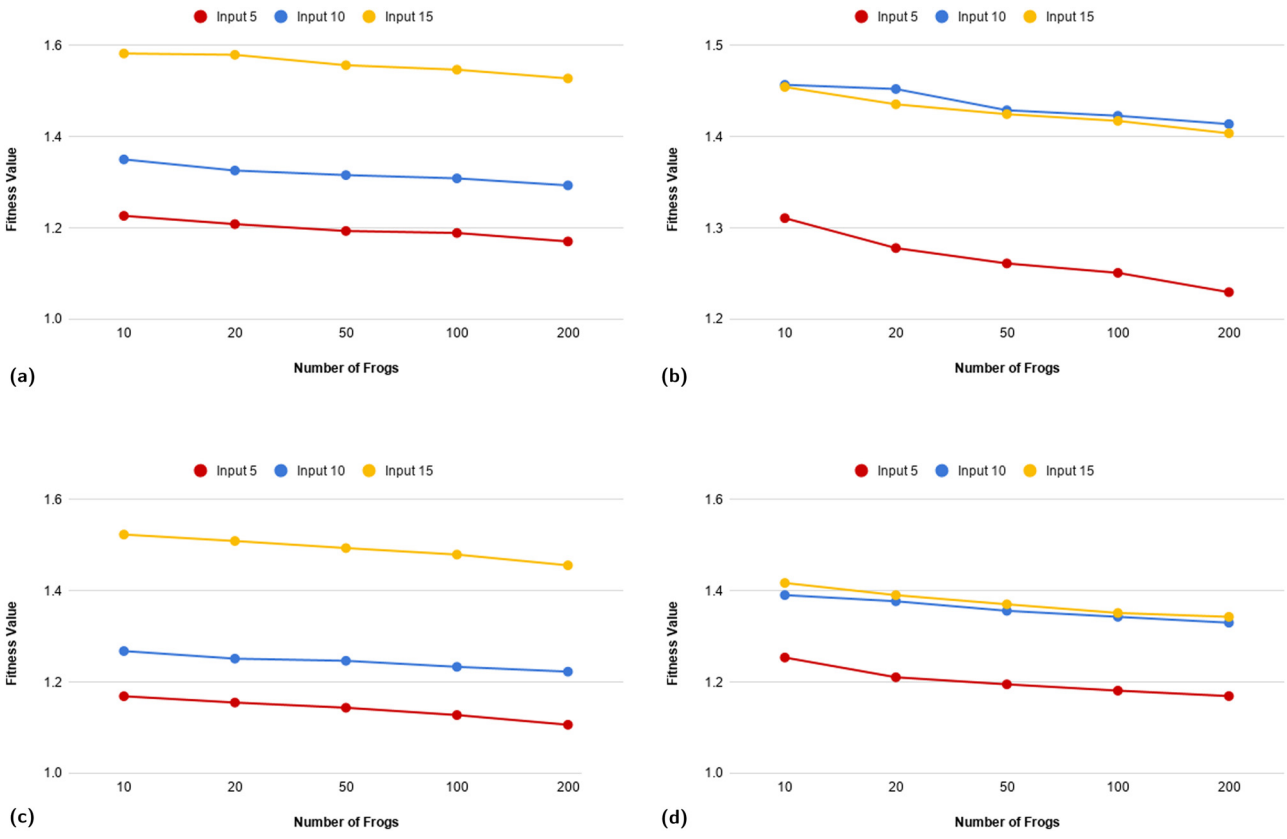

Figure 23: Effect of frog population size on performance of ISFLA and proposed ESFLA-RW. (a) ISFLA – equal Load, (b) ISFLA – unequal load, (c) ESFLA-RW – equal load, (d) ESFLA-RW – unequal load.

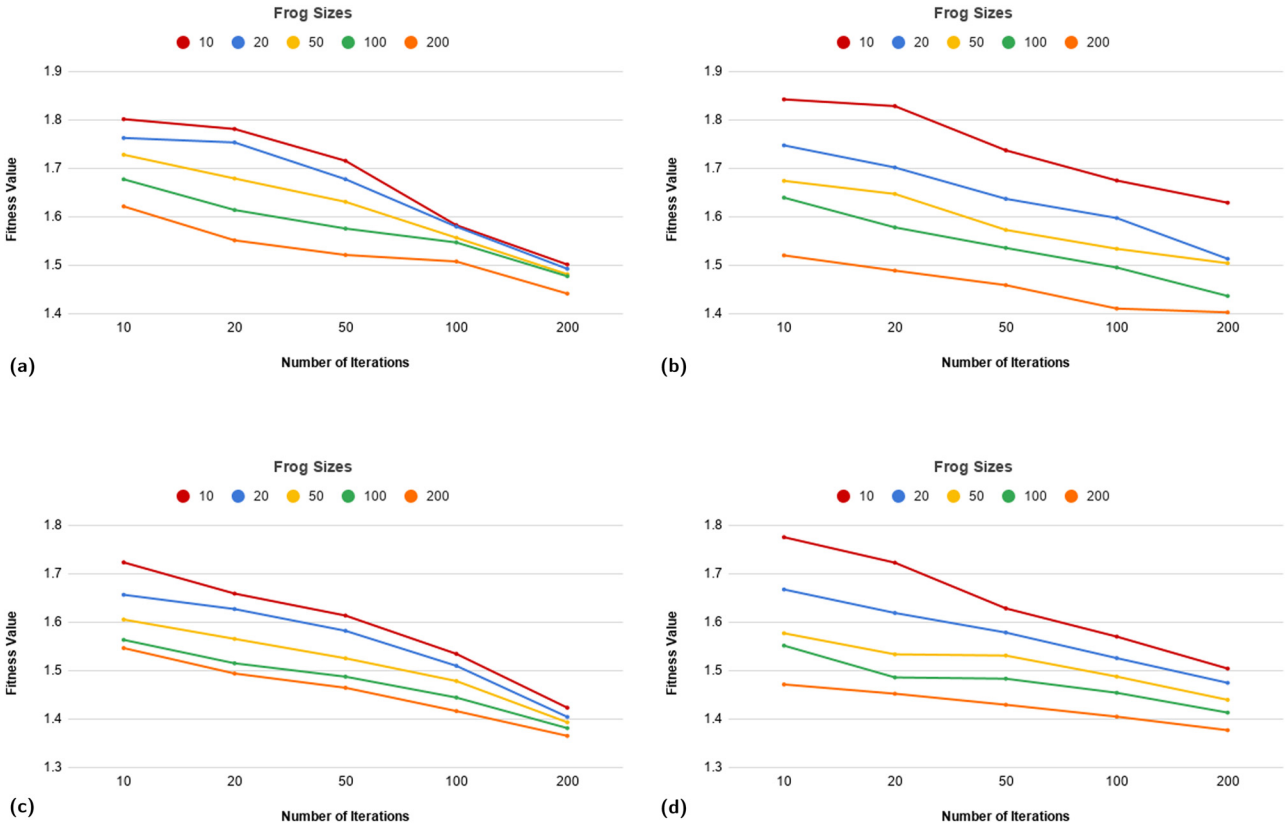

Figure 24: Effect of frog population size on convergence of ISFLA and proposed ESFLA-RW. (a) ISFLA – equal load, (b) ISFLA – unequal load, (c) ESFLA-RW – equal load, (d) ESFLA-RW – unequal load.

algorithms. Experiments are performed on large input dataset (input 15). The best fitness value obtained from ESFLA-RW by varying the frog population sizes is measured. The obtained results, shown in Figure 24, indicates that frog population size has an impact on the convergence rate of algorithms. Performance of population-based heuristic algorithms depends on the population diversity. Large population diversity leads to better exploration of search space. The obtained results validate the impact of frog population size on performance of SFLA.

5.3.2 Effect of probability of information exchange

This section presents the impact of probability of information exchange on performance of ISFLA and ESFLA-RW. A random value between 0 to 1 is generated for each local exploration, if the generated value is less than probability of information exchange then information exchange is carried out to improve frog, otherwise the worst frog improvement is skipped. This concept is similar to crossover rate in genetic algorithms.

Experiments are performed for two input datasets from each category of small, medium and large datasets. Input 1 and input 5 from a small dataset, input 6 and input 10 from medium dataset, while input 11 and input 15 from large dataset are taken. The probability of information exchange is tuned from 0.1 to 1.0 with a step size of 0.1. The best fitness values obtained for various probability of information exchange are noted down. The obtained results are shown in Figure 25. It is observed from the graphs that to achieve better fitness, probability of information exchange should be 1, which means information exchange should always be carried out to improve local exploration and improve the final results.

5.3.3 Effect of probability of energy-biased load reduction phase

This section presents the impact of probability of energy-biased load reduction phase on performance of ISFLA and ESFLA-RW. A random value between 0 and 1 is generated for each local exploration; if the generated value is less than the probability of energy-biased load reduction,

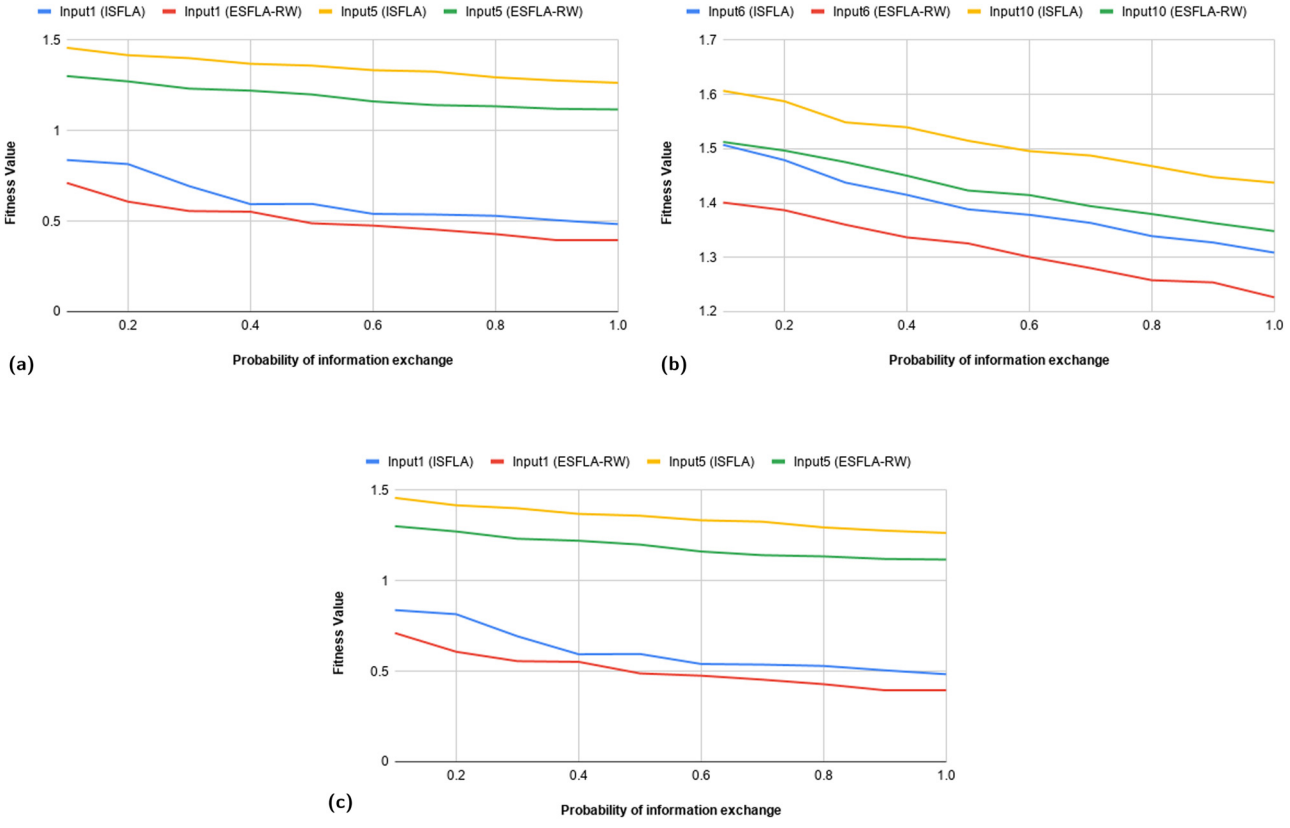

Figure 25: Effect of probability of information exchange on performance of ISFLA and ESFLA-RW. (a) Small dataset, (b) medium dataset, (c) large dataset.

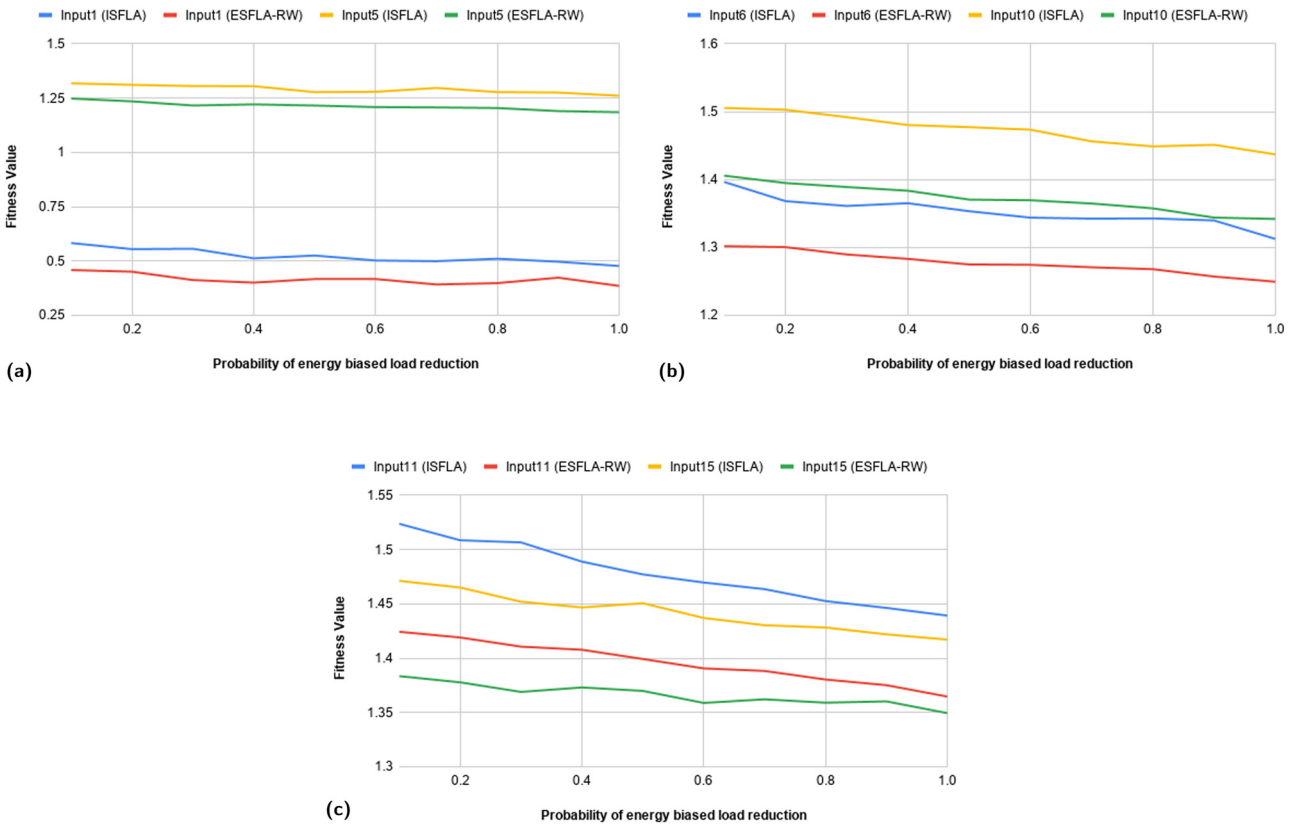

Figure 26: Effect of probability of energy-biased load reduction phase. (a) Small dataset, (b) medium dataset, (c) large dataset.

then load reduction phase is carried out to improve frog, otherwise the phase is skipped. This concept is similar to mutation rate in genetic algorithms.

Experiments are performed for two input datasets from each category of small, medium and large datasets. Input 1 and input 5 are small datasets, input 6 and input 10 are medium datasets and input 11 and input 15 are large datasets. The probability of energy-biased load reduction phase is tuned from 0.1 to 1.0 with a step size of 0.1. The best fitness values obtained for various probability of energy-biased load reduction phases are noted down. The obtained results are shown in Figure 26. It is observed from the graphs that the effect of probability of energy-biased load reduction phase is small as compared to the effect of probability of information exchange on these algorithms.

6 Conclusions

This article presented an ESFLA, which is an extension of the ISFLA presented by Edla et al. [19] for load balancing of gateways in WSNs. This article presented three variations to improve the local exploration of the SFLA. First variation (ESFLA-GG) replaces the worst frogs of trailing half sub-memeplexes with the global best frog. Second variation (ESFLA-SS) used a probability-based stochastic sampling operator to select source and target frogs. Third variation (ESFLA-RW) used a probabilistic RW operator to select source frog, and the worst frog of sub-memeplex is a default target frog. RW operator in the local exploration phase gives wide exposure to choose the frog for information exchange rather than always sticking to the best frog. Also, the article presented a novel energy-biased load reduction phase. This phase focuses on improving the survival of gateways with lowest residual energy by reducing its load. This leads to load balancing and increase in the lifetime of the gateways. We have compared the proposed variations of ESFLA with ISFLA presented by Edla et al. [19] for multiple datasets of varying scenarios of WSNs. The experimental results show that the proposed ESFLA-RW performs better than ISFLA in terms of load balancing, best solution, standard deviation, network energy consumption, network lifetime and convergence rate. Effect of algorithmic parameters on performance of ISFLA and ESFLA-RW is investigated. Large frog population size improves

the chances of diversity and exploration, which lead to better results and fast convergence. Results show that higher probability of information exchange and energy-biased load reduction phase gives better solution for ISFLA and ESFLA-RW.

Edla et al. [19] reported that the ISFLA algorithm is better when compared with other load balancing techniques, namely NLDLB, SBLB, SGA and NGA. Results show that the proposed ESFLA-RW is better than ISFLA for load balancing of gateways in WSNs in terms of the several parameters. Although there is one potential limitation that the execution time of ESFLA-RW is slightly higher than ISFLA due to the introduction of an additional RW selection phase. There is scope to investigate the performance of proposed variations to mathematical benchmark functions with higher dimensions. The future work is to improve the fitness function considering the energy required by gateways to aggregate data before sending them to the base station.

Conflict of interest: Authors state no conflict of interest.

Data availability statement: All data generated or analysed during this study are included in this published article.

References

[1] I. F. Akyildiz, W. Su, Y. Sankarasubramaniam, and E. Cayirci, "Wireless sensor networks: a survey," *Computer networks*, vol. 38, no. 4, pp. 393–422, 2002.

[2] J. L. Burbank, P. F. Chimento, B. K. Haberman, and W. T. Kasch, "Key challenges of military tactical networking and the elusive promise of MANET technology," *IEEE Commun. Mag.*, vol. 44, no. 11, pp. 39–45, 2006.

[3] J. Ko, C. Lu, M. B. Srivastava, J. A. Stankovic, A. Terzis, M. Welsh, "Wireless sensor networks for healthcare," *Proc. IEEE*, vol. 98, no. 11, pp. 1947–1960, November 2010.

[4] H. J. Korber, H. Wattar, and G. Scholl, "Modular wireless real-time sensor/actuator network for factory automation applications," *IEEE Trans. Indust. Inform.*, vol. 3, no. 2, pp. 111–119, 2007.

[5] O. Palagin, V. Romanov, I. Galelyuka, O. Voronenko, D. Artemenko, O. Kovyrova, and Y. Sarakhan, "Computer devices and mobile information technology for precision farming," In: *2013 IEEE 7th International Conference on Intelligent Data Acquisition and Advanced Computing Systems* (*IDAACS*), IEEE, vol. 1, September 2013, pp. 47–51.

[6] G. Anastasi, M. Conti, M. DiFrancesco, and A. Passarella, "Energy conservation in wireless sensor networks: A survey," *Ad hoc networks*, vol. 7, no. 3, pp. 537–568, 2009.

[7] C. Y. Chong and S. P. Kumar, "Sensor networks: evolution, opportunities, and challenges," *Proc. IEEE*, vol. 91, no. 8, pp. 1247–1256, 2003.

[8] Y. K. Yousif, R. Badlishah, N. Yaakob, and A. Amir, "An energy efficient and load balancing clustering scheme for wireless sensor network (WSN) based on distributed approach," *J. Phys.: Conf. Ser.*, vol. 1019, no. 1, p. 012007, June 2018, IOP Publishing.

[9] C. P. Low, C. Fang, J. M. Ng, and Y. H. Ang, "Efficient load-balanced clustering algorithms for wireless sensor networks," *Comput. Commun.*, vol. 31, no. 4, pp. 750–759, 2008.

[10] P. Kuila and P. K. Jana, "A novel differential evolution based clustering algorithm for wireless sensor networks," *Appl. Soft Comput.*, vol. 25, pp. 414–425, 2014.

[11] S. Mor and M. V. Saroha, "Load balancing in wireless sensor networks," *Int. J. Softw. Web Sci.*, vol. 4, no. 2, pp. 116–119, 2013.

[12] G. Gupta and M. Younis, "Load-balanced clustering of wireless sensor networks," In: *IEEE International Conference on Communications, 2003. ICC'03.* (*Vol. 3*), IEEE, May 2003, pp. 1848–1852.

[13] P. Kuila and P. K. Jana, "Energy efficient clustering and routing algorithms for wireless sensor networks: Particle swarm optimization approach," *Eng. Appl. Artif. Intel.*, vol. 33, pp. 127–140, 2014.

[14] P. Kuila and P. K. Jana, "Energy efficient load-balanced clustering algorithm for wireless sensor networks," *Proc. Tech.*, vol. 6, pp. 771–777, 2012.

[15] F. Fanian and M. K. Rafsanjani, "Memetic fuzzy clustering protocol for wireless sensor networks: Shuffled frog leaping algorithm," *Appl. Soft Comput.*, vol. 71, 568–590, 2018.

[16] Y. Liao, H. Qi, and W. Li, "Load-balanced clustering algorithm with distributed self-organization for wireless sensor networks," *IEEE Sensors J.*, vol. 13, no. 5, pp. 1498–1506, 2012.

[17] J. S Leu, T. H. Chiang, M. C. Yu, and K. W. Su, "Energy efficient clustering scheme for prolonging the lifetime of wireless sensor network with isolated nodes," *IEEE Commun. Lett.*, vol. 19, no. 2, pp. 259–262, 2014.

[18] V. S. Gattani and S. H. Jafri, "Data collection using score-based load balancing algorithm in wireless sensor networks," In: *2016 International Conference on Computing Technologies and Intelligent Data Engineering* (*ICCTIDE'16*), IEEE, January 2016, pp. 1–3.

[19] D. R. Edla, A. Lipare, R. Cheruku, and V. Kuppili, "An efficient load balancing of gateways using improved shuffled frog leaping algorithm and novel fitness function for WSNs," *IEEE Sensors J.*, vol. 17, no. 20, pp. 6724–6733, 2017.

[20] P. S. Rao and H. Banka, "Energy efficient clustering algorithms for wireless sensor networks: novel chemical reaction optimization approach," *Wirel. Netw.*, vol. 23, no. 2, pp. 433–452, 2017.

[21] D. R. Edla, V. Deshmukh, R. Cheruku, S. D. Saheeka, and B. Yadav, "A novel green stable evolutionary routing algorithm for energy efficiency in WSNS," In: *2017 International Conference on Advances in Computing, Communications and Informatics* (*ICACCI*), IEEE, September 2017, pp. 724–728.

[22] P. K. Agarwal and C. M. Procopiuc, "Exact and approximation algorithms for clustering," *Algorithmica*, vol. 33, no. 2, pp. 201–226, 2002.

[23] M. Rout and K. M. Koudjonou, "An evolutionary algorithm based hybrid parallel framework for Asia foreign exchange rate prediction," *Nature Inspired Computing for Data Science*, Cham: Springer, 2020, pp. 279–295.
[24] J. R. S Iruela, L. G. B. Ruiz, M. C. Pegalajar, and M. I. Capel, "A parallel solution with GPU technology to predict energy consumption in spatially distributed buildings using evolutionary optimization and artificial neural networks," *Energy Convers. Manag.*, vol. 207, p. 112535, 2020.
[25] S. Pulipati and M. Ramakrishnan, "Topological and Attribute Link Prediction using Firefly algorithm," *Open Comput. Sci.*, vol. 10, no. 1, pp. 33–41, 2020.
[26] P. Kaur and M. Sharma, "Diagnosis of human psychological disorders using supervised learning and nature-inspired computing techniques: a meta-analysis," *J. Med. Syst.*, vol. 43, no. 7, pp. 204, 2019.
[27] M. Sharma, G. Singh, R. Singh, and G. Singh, "Analysis of DSS queries using entropy based restricted genetic algorithm," *Appl. Math. Inform. Sci.*, vol. 9, no. 5, pp. 2599, 2015.
[28] M. Sharma and P. Kaur, "A comprehensive analysis of nature-inspired meta-heuristic techniques for feature selection problem," *Arch. Comput. Methods Eng.*, pp. 1–25, 2020.
[29] S. C. Satapathy, A. Naik, and K. Parvathi, "Rough set and teaching learning based optimization technique for optimal features selection," *Cent. Eur. J. Comput. Sci.*, vol. 3, no. 1, pp. 27–42, 2013.
[30] A. Alarifi, A. Tolba, Z. Al-Makhadmeh, and W. Said, "A big data approach to sentiment analysis using greedy feature selection with cat swarm optimization-based long short-term memory neural networks," *J. Supercomputing*, vol. 76, no. 6, pp. 4414–4429, 2020.
[31] S. Salcedo-Sanz, B. Saavedra-Moreno, A. Paniagua-Tineo, L. Prieto, and A. Portilla-Figueras, "A review of recent evolutionary computation-based techniques in wind turbines layout optimization problems," *Open Comput. Sci.*, vol. 1, no. 1, pp. 101–107, 2011.
[32] D. R. Edla, M. C. Kongara, and R. Cheruku, "SCE-PSO based clustering approach for load balancing of gateways in wireless sensor networks," *Wirel. Netw.*, vol. 25, no. 3, pp. 1067–1081, 2019.
[33] N. A. Latiff, C. C. Tsimenidis, and B. S. Sharif, "Energy-aware clustering for wireless sensor networks using particle swarm optimization," In: *2007 IEEE 18th International Symposium on Personal, Indoor and Mobile Radio Communications*, IEEE, September 2007, pp. 1–5.
[34] M. Eusuff, K. Lansey, and F. Pasha, "Shuffled frog-leaping algorithm: a memetic meta-heuristic for discrete optimization," *Eng. Optim.*, vol. 38, no. 2, pp. 129–154, 2006.
[35] G. Y. Zhu and W. B. Zhang, "An improved shuffled frog-leaping algorithm to optimize component pick-and-place sequencing optimization problem," *Expert Syst. Appl.*, vol. 41, no. 15, pp. 6818–6829, 2014.
[36] K. K. Bhattacharjee and S. P. Sarmah, "Shuffled frog leaping algorithm and its application to 0/1 knapsack problem," *Appl. Soft Comput.*, vol. 19, pp. 252–263, 2014.
[37] X. H. Luo, Y. Yang, and X. Li, "Solving TSP with shuffled frog-leaping algorithm," In: *2008 Eighth International Conference on Intelligent Systems Design and Applications*, IEEE, vol. 3, November 2008, pp. 228–232.
[38] C. Fang and L. Wang, "An effective shuffled frog-leaping algorithm for resource-constrained project scheduling problem," *Comput. Operat. Res.*, vol. 39, no. 5, pp. 890–901, 2012.
[39] D. Lei and X. Guo, "A shuffled frog-leaping algorithm for hybrid flow shop scheduling with two agents," *Expert Syst. Appl.*, vol. 42, no. 23, pp. 9333–9339, 2015.
[40] J. Cai, R. Zhou, and D. Lei, "Dynamic shuffled frog-leaping algorithm for distributed hybrid flow shop scheduling with multiprocessor tasks," *Eng. Appl. Artif. Intel.*, vol. 90, 103540, 2020.
[41] P. Kaur and S. Mehta, "Resource provisioning and work flow scheduling in clouds using augmented shuffled frog leaping algorithm," *J. Parallel Distr. Comput.*, vol. 101, pp. 41–50, 2017.
[42] J. Tang, R. Zhang, P. Wang, Z. Zhao, L. Fan, and X. Liu, "A discrete shuffled frog-leaping algorithm to identify influential nodes for influence maximization in social networks," *Knowledge-Based Syst.*, vol. 187, p. 104833, 2020.
[43] J. Luo and M. R. Chen, "Improved shuffled frog leaping algorithm and its multi-phase model for multi-depot vehicle routing problem," *Expert Syst. Appl.*, vol. 41, no. 5, pp. 2535–2545, 2014.
[44] P. Roy, P. Roy, and A. Chakrabarti, "Modified shuffled frog leaping algorithm with genetic algorithm crossover for solving economic load dispatch problem with valve-point effect," *Appl. Soft Comput.*, vol. 13, no. 11, pp. 4244–4252, 2013.
[45] S. Sharma, T. K. Sharma, M. Pant, J. Rajpurohit, and B. Naruka, "Centroid mutation-embedded shuffled frog-leaping algorithm," *Proc. Comput. Sci.*, vol. 46, pp. 127–134, 2015.
[46] P. Sharma, N. Sharma, and H. Sharma, "Binomial crossover-embedded shuffled frog leaping algorithm," In: *2016 International Conference on Computing, Communication and Automation* (*ICCCA*), IEEE, April 2016, pp. 321–326.
[47] P. Sharma, N. Sharma, and H. Sharma, "Elitism based shuffled frog leaping algorithm," In: *2016 International Conference on Advances in Computing, Communications and Informatics* (*ICACCI*), IEEE, September 2016, pp. 788–794.
[48] H. Wang, X. Zhen, and X. Tu, "SFDE: Shuffled frog-leaping differential evolution and its application on cognitive radio throughput," *Wirel. Commun. Mob. Comput.*, vol. 2019, 2019.
[49] J. Zhang and T. Yang, "Clustering model based on node local density load balancing of wireless sensor network," In *2013 Fourth International Conference on Emerging Intelligent Data and Web Technologies*, IEEE, September 2013, pp. 273–276.
[50] S. Hussain, A. W. Matin, and O. Islam, "Genetic algorithm for hierarchical wireless sensor networks," *J. Netw.*, vol. 2, no. 5, pp. 87–97, 2007.
[51] P. Kuila, S. K. Gupta, and P. K. Jana, "A novel evolutionary approach for load balanced clustering problem for wireless sensor networks," *Swarm Evolut. Comput.*, vol. 12, pp. 48–56, 2013.
[52] W. R. Heinzelman, A. Chandrakasan, and H. Balakrishnan, "Energy-efficient communication protocol for wireless microsensor networks," In: *Proceedings of the 33rd annual Hawaii International Conference on System Sciences*, IEEE, January 2000, p. 10.
[53] W. B. Heinzelman, A. P. Chandrakasan, and H. Balakrishnan, "An application-specific protocol architecture for wireless

microsensor networks," *IEEE Trans. Wirel. Commun.*, vol. 1, no. 4, pp. 660–670, 2002.

[54] S. Lindsey and C. S. Raghavendra, "PEGASIS: Power-efficient gathering in sensor information systems," In: *Proceedings, IEEE Aerospace Conference*, IEEE, vol. 3, March 2002, p. 33.

[55] O. Younis and S. Fahmy, "Distributed clustering in ad-hoc sensor networks: A hybrid, energy-efficient approach," In: *IEEE INFOCOM 2004*, IEEE, vol. 1, 2004.

[56] J. Tillett, R. Rao, and F. Sahin, "Cluster-head identification in ad hoc sensor networks using particle swarm optimization," In: *2002 IEEE International Conference on Personal Wireless Communications*, IEEE, December 2002, pp. 201–205.

[57] S. M. Guru, S. K. Halgamuge, and S. Fernando, "Particle swarm optimisers for cluster formation in wireless sensor networks," In: *2005 International Conference on Intelligent Sensors, Sensor Networks and Information Processing*, IEEE, December 2005, pp. 319–324.

[58] B. Singh and D. K. Lobiyal, "A novel energy-aware cluster head selection based on particle swarm optimization for wireless sensor networks," *Human-Centric Comput. Inform. Sci.*, vol. 2, no. 1, pp. 13, 2012.

[59] M. M. Eusuff and K. E. Lansey, "Optimization of water distribution network design using the shuffled frog leaping algorithm," *J. Water Resour. Plan. Manag.*, vol. 129, no. 3, pp. 210–225, 2003.

[60] J. E. Baker, "Reducing bias and inefficiency in the selection algorithm," In: *Proceedings of the Second International Conference on Genetic Algorithms*, vol. 206, July 1987, pp. 14–21.

[61] T. Pencheva, K. Atanassov, and A. Shannon, "Modelling of a roulette wheel selection operator in genetic algorithms using generalized nets," *Int. J. Bioautomation*, vol. 13, no. 4, pp. 257–264, 2009.

[62] P. N. Suganthan, N. Hansen, J. J. Liang, K. Deb, Y. P. Chen, A. Auger, and S. Tiwari, "Problem definitions and evaluation criteria for the CEC 2005 special session on real-parameter optimization," KanGAL Report, 2005005(2005), 2005.