Optimal LEACH Protocol with Improved Bat Algorithm in Wireless Sensor Networks

Xingjuan Cai[1], Youqiang Sun[1], Zhihua Cui[1*], Wensheng Zhang[2] and Jinjun Chen[3]
[1] Complex System and Computational Intelligence Laboratory, Taiyuan University of Science and Technology
Taiyuan, Shanxi, 030024, China
[e-mail: xingjuancai@163.com]
[e-mail: youqiang_sun@163.com]
[e-mail: zhihua.cui@hotmail.com]
[2] State Key Laboratory of Intelligent Control and Management of Complex Systems, Institute of Automation
Chinese Academy of Sciences
Beijing, 100190, China
[e-mail: wensheng.zhang@ia.ac.cn]
[3] University of Technology Sydney
Sydney, NSW 2007, Australia
[e-mail: jinjun.chen@gmail.com]
*Corresponding author: Zhihua Cui

Received August 30, 2018; revised October 16, 2018; revised November 14, 2018; accepted November 21, 2018; published May 31, 2019

Abstract

A low-energy adaptive clustering hierarchy (LEACH) protocol is a low-power adaptive cluster routing protocol which was proposed by MIT's Chandrakasan for sensor networks. In the LEACH protocol, the selection mode of cluster-head nodes is a random selection of cycles, which may result in uneven distribution of nodal energy and reduce the lifetime of the entire network. Hence, we propose a new selection method to enhance the lifetime of network, in this selection function, the energy consumed between nodes in the clusters and the power consumed by the transfer between the cluster head and the base station are considered at the same time. Meanwhile, the improved FTBA algorithm integrating the curve strategy is proposed to enhance local and global search capabilities. Then we combine the improved BA with LEACH, and use the intelligent algorithm to select the cluster head. Experiment results show that the improved BA has stronger optimization ability than other optimization algorithms, which the method we proposed (FTBA-TC-LEACH) is superior than the LEACH and LEACH with standard BA (SBA-LEACH). The FTBA-TC-LEACH can obviously reduce network energy consumption and enhance the lifetime of wireless sensor networks (WSNs).

Keywords: Low-energy adaptive clustering hierarchy (LEACH) protocol, Bat algorithm, Curve decline strategy, Triangle-flipping strategy

1. Introduction

As a prerequisite for the development of the Internet of Things (IoT), technologies related to object recognition and information acquisition are of paramount importance. Sensor technology is therefore indispensable. WSNs constitute an important part of IoT, which is often used for special purposes such as location, early warning systems, and emergency rescue [1]. These sensors must be able to collect [2], process, and transmit information. Desired information can be collected and detected by deploying sufficient sensors in an area to be monitored. WSNs are becoming widely used given IoT proliferation, such as in smart cities, intelligent transportation, military defense, modern agriculture, health care, and environmental monitoring applications [2-4].

Due to the limited energy of sensor nodes, extending their use as much as possible is important when studying WSNs. Routing protocol technology greatly affects the network life cycle [5]. Hierarchical routing is an efficient technique to reduce energy consumption via data aggregation and fusion by using a method to decrease the number of transmissions to the base station [6]. The LEACH is the first and most well-known hierarchical routing protocol [7]. In a typical hierarchical routing algorithm, the clustering idea in LEACH has been referenced in many subsequent routing protocols, including hierarchical routing protocols such as the threshold sensitive energy efficient sensor network protocol [7, 8] and power-efficient gathering in sensor information systems [9]. The clustering method reduces network energy consumption and improves network lifetime. The core concept of the LEACH is based on rounds, and the basic idea of the protocol is to randomly select a cluster-head node cyclically and distribute the network energy load evenly among sensor nodes [10]. This approach enhances energy consumption because transmissions are only managed by cluster-head nodes rather than all sensor nodes. However, the LEACH protocol does not consider current node energy, and the selection of random cluster-heads can easily result in uneven energy consumption among network nodes, which shortens the network life cycle. A monitoring blind spot can then form in the rear stage of the network and affect overall network performance [11, 12].

Many LEACH improvements have been implemented[13], many of which reduce energy consumption in the LEACH protocol; others consider the energy consumption balance. Most earlier improvements were based on head node selection. For example, LI Yan proposed a new modified protocol using a cluster-head multi-hop algorithm based on LEACH in which an optimal path forms between cluster-heads leading to the base station [14]. A simulated annealing algorithm has also been used to select a cluster-head node when implementing the LEACH protocol in WSNs, demonstrating much better performance compared with the original protocol [15]. An improved routing protocol named ANT-LEACH was proposed by WANG Lin, who adopted the strategy of combining an ant colony optimization algorithm and the routing process of cluster-head nodes [16]. HAN Dong-xue put forth a PSO-based double cluster-heads clustering algorithm; the master cluster-head and vice cluster-head assume different roles, where the master cluster-head completes data collection and date integration while the vice cluster-head is involved in communication with the base station [17].

Despite these achievements, most methods are based on a predefined searching pattern or deterministic optimization algorithms. To identify a better energy-saving solution, superior optimization of cluster-head node selection is required. Cluster-head node selection is a complex optimization problem that should consider the node location and node residual

energy. A biological heuristic algorithm is an effective method for solving this optimization problem, such as via standard particle swarm optimization (PSO) [18-21], ant colony optimization [22-27], cuckoo search (CS) [28-32], and artificial bee colony algorithm (ABC) [33-38].

In 2010, University of Cambridge scholar YANG proposed a new intelligent optimization algorithm, the bat algorithm (BA), by simulating the echolocation behavior of bats [39-42]. The algorithm is a population-based stochastic optimization algorithm of which bat individuals are the basic units. The movement of the entire population produces an evolutionary process from disorder to order in the problem-solving space, thereby obtaining the optimal solution. Li and Zhou [43] proposed a complex-valued BA where each bat is coded with a complex number. Cai and Wang [44] presented a fast BA with triangle-flipping strategy (FTBA), which increased the global search ability of the algorithm. This paper continues to improve FTBA coupled with another curve strategy to enhance local optimization.

In this paper, an FTBA-TC-LEACH routing algorithm based on BA is proposed, and a new cluster-head selection mode is adopted to solve the problem of uneven cluster-head distribution in LEACH, which causes non-uniform energy consumption. The rest of this paper is organized as follows: in Section 2, we review the background of the LEACH protocol and propose improved cluster-head selection. Section 3 introduces the modified BA and uses it for cluster-head optimization. Section 4, we verify the proposed improvements through simulation experiments. Section 5 presents conclusions and a brief overview of our future work.

2. Background

2.1 Low-Energy Adaptive Clustering Hierarchy (LEACH)

The LEACH protocol is a typical hierarchical routing protocol algorithm that is a commonly used clustering routing protocol; therefore, LEACH was taken as the research object given high representativeness. Assumed preconditions of the LEACH protocol are as follows:

(1) All sensor nodes are the same, and the radio signal in each direction consumes the same energy; the initial energy of each node is equal, the energy is limited, and each node can perceive its remaining energy. The node has power control to change the transmit power and control the transfer distance, and each node has sufficient computing power;

(2) all nodes can communicate directly with each other, which also applies to nodes and the base station;

(3) the sink nodes are fixed and far from the entire WSN. Energy consumption of the sink nodes is not considered in this study, as they are assumed to have sufficient energy supply; and

(4) the nodes are stationary.

The WSN is assumed to be

$$S = \{Sensor_1, Sensor_2, \cdots, Sensor_v\} \tag{1}$$

The behavior of each sensor $Sensor_i$ in the LEACH protocol can be described as in Eq. (2):

$$Sensor_i = (Time, Head, Table, Communication) \tag{2}$$

In the $Sensor_i$,

Time indicates a stable time slice, during which $Sensor_i$ is in accordance with the current state; if it exceeds the time segment, then the sensor will consider adjusting the current state;

Head indicates if the sensor is a cluster-head node;

Table: If $Sensor_i$ is cluster-head node, then *Table* is used to indicate position information of the remaining sensors in the cluster;

Communication: If $Sensor_i$ is cluster-head node, then *Communication* reflects communication between the sensor and the base station and other sensors in the cluster; if $Sensor_i$ is not a cluster-head node, it only indicates communication between the sensor and corresponding cluster-head node.

The LEACH protocol cycles through a cluster reconstruction process wherein each phase of the cycle can be described as a round. Each round is divided into two phases: establishment phase of the cluster (setup phase) and stable phase of the transmitted data (ready phase). To conserve resource overhead, the ready phase lasts longer than the setup phase. **Fig. 1** shows the LEACH protocol operation cycle chart.

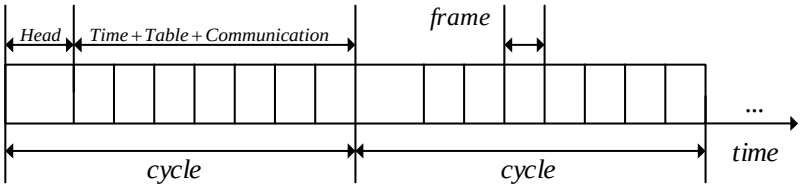

Fig. 1. LEACH protocol operation cycle diagram

2.2 Original Cluster-head Node Selection in LEACH

In the setup phase, cluster-head node selection is based on the total number of cluster-head nodes required in the network and the number of times each node has become the cluster-head node so far. Options are that each sensor node randomly selects a value in [0, 1]; if the selected value is less than a certain threshold $T(n)$, then the sensor node will be a cluster-head node. The calculation method of $T(n)$ is as follows:

$$T(n)=\begin{cases}\dfrac{P}{1-P\times[r*\bmod(\dfrac{1}{P})]} & n\in G\\ 0 & n\notin G\end{cases} \tag{3}$$

$$P=\frac{l}{k} \tag{4}$$

In Eq. (4), P denotes a percentage of the cluster-head node, l denotes the cluster-head nodes needed in the network, and k denotes the total number of sensor nodes. In Eq. (3), r is the current number of rounds; G is a set; $1/P$ round has not been selected as a cluster-head node in the past; and mod is the modulo operator. After selecting the cluster-head node, the entire network is informed by broadcast.

In the ready phase, sensor nodes transmit gathered data to the cluster-head node. The cluster-head node collects data from all nodes in the cluster and then transmits it to the base station.

2.3 Improved Selection of Cluster-head Nodes

To ensure that each sensor node works as long as possible, after each cluster-head operating time *Time*, the cluster-head must be reselected, and the remaining sensors can be classified according to the cluster-head so each class (i.e., each cluster) contains only one cluster-head. Therefore, the choice of the cluster-head node will directly affect the performance of the algorithm. Cluster-head nodes tend to be randomly generated; however, this method can easily lead to the following adverse consequences:

(1) Intra-cluster nodes become unevenly distributed near the cluster-head node (i.e., the distance variance between cluster nodes and the cluster-head node is larger), resulting in greater energy consumption; and

(2) a long distance from the cluster-head node to the base station will affect the lifetime of the cluster-head.

To avoid these defects, we introduce an intelligent optimization algorithm into cluster-head selection so the two defects can be reduced as much as possible to minimize energy consumption. In other words, we use an intelligent optimization algorithm to optimize the objective function of cluster C_j as follows:

$$\min \quad f = \alpha_D \cdot std(C_j) + (1-\alpha_D) \cdot D_j \tag{5}$$

$$std(C_j) = \frac{\sum_{k=1}^{L_j} D_{jk}^2}{L_j} - (\frac{\sum_{k=1}^{L_j} D_{jk}}{L_j})^2 \tag{6}$$

$$E(Ave_j) = \frac{\sum_{i=1}^{k} E(C_{ji})}{k} \tag{7}$$

In Eq. (5), $\alpha_D \in (0,1)$ is the weight, set at 0.8 in this experiment; and D_j is the Euclidean distance from the base station to the cluster-head node of cluster C_j. If L_j indicates the number of sensors contained in cluster C_j, then D_{jk} represents Euclidean distance between the k sensor and the cluster-head in cluster C_j, and $std(C_j)$ represents the distance variance within the cluster C_j. $std(C_j)$ is explained in Eq. (6).

In addition, to avoid premature energy depletion at some nodes, the cluster-head position is guaranteed to be larger than the average energy $E(Ave_j)$ within the cluster. The average energy is calculated by Eq. (7), where k is the number of nodes in cluster C_j, and $E(C_{ji})$ is the residual energy of node i in cluster C_j.

In the following parts, we use the modified intelligence algorithm to solve the identified optimization problems; the fitness function is shown above.

3. Proposed FTBA-TC-LEACH Routing Algorithm

3.1 Standard Bat Algorithm

BA is a heuristic intelligent algorithm that simulates the principle of echolocation used in bat

predation. BA has passed standard test functions and achieved good results for solving continuity optimization problems [45]. The single-objective unconstrained optimization problem is considered in this paper as Eq. (8):

$$\min \ f(\boldsymbol{x}), \ [\boldsymbol{x}=(x_1, x_2, ..., x_k, ..., x_D) \in E] \tag{8}$$

Suppose there are n virtual bats, and the ith bat: $(i = 1, 2, 3, ..., N)$ is represented as Eq. (9):

$$< \boldsymbol{x}_i(t), \boldsymbol{v}_i(t), fr_i(t), A_i(t), r_i(t) > \tag{9}$$

where $\boldsymbol{x}_i(t) = (x_{i1}(t), x_{i2}(t), ..., x_{ik}(t), ..., x_{iD}(t))$ and $\boldsymbol{v}_i(t) = (v_{i1}(t), v_{i2}(t), ..., v_{ik}(t), ..., v_{iD}(t))$ are the position and velocity of the ith bat in generation t, respectively, with frequency $fr_i(t)$, loudness $A_i(t)$, and emission rate $r_i(t)$ as the three required parameters.

In the next generation, the velocity is updated as follows:

$$v_{ik}(t+1) = v_{ik}(t) + (x_{ik}(t) - p_k(t)) \cdot fr_i(t), \tag{10}$$

where $\boldsymbol{p}(t) = (p_1(t), p_2(t), ..., p_k(t), ..., p_D(t))$ is the best position found thus far by the entire swarm. Eq. (10) can be viewed as a combination of the inertia $v_{ik}(t)$ and the influence of $\boldsymbol{p}(t)$. The frequency $fr_i(t)$ is calculated as follows:

$$fr_i(t) = fr_{\min} + (fr_{\max} - fr_{\min}) \cdot rand_1, \tag{11}$$

where $fr_{\max}$ and $fr_{\min}$ are the maximum and minimum frequency values, respectively, and $rand_1$ is a random number uniformly distributed within [0,1].

To reflect the bat decision, the position changes with some randomness. Let $rand_2$ be a random number uniformly distributed within [0, 1]; if $rand_2 < r_i(t)$ is satisfied, then the ith bat will execute the following global search pattern:

$$x'_{ik}(t+1) = x_{ik}(t) + v_{ik}(t+1). \tag{12}$$

Otherwise, the following local search pattern is adopted:

$$x'_{ik}(t+1) = p_k(t) + \varepsilon_{ik} \cdot \bar{A}(t), \tag{13}$$

where ε_{ik} is a random number generated by a uniform distribution within [–1, 1], $\bar{A}(t)$ is the average loudness of all bats, and

$$\bar{A}(t) = \frac{\sum_{i=1}^{n} A_i(t)}{n} \tag{14}$$

After the $\boldsymbol{x}'_{\boldsymbol{ik}}(t+1) = (x'_{i1}(t+1), x'_{i2}(t+1), ..., x'_{ik}(t+1), ..., x'_{iD}(t+1))$ is obtained by Eq. (12) and Eq. (13), the new $\boldsymbol{x}_i(t+1)$ can be updated as follows:

$$\boldsymbol{x}_i(t+1) = \begin{cases} \boldsymbol{x}'_i(t+1) & if \ rand_3 < A_i(t) \ and \ f(\boldsymbol{x}'_i(t+1)) < f(\boldsymbol{x}_i(t)) \\ \boldsymbol{x}_i(t) & otherwise \end{cases} \tag{15}$$

where $rand_3$ is a random number generated by uniform distribution within [0 , 1]. Similar to CS, Eq. (15) implies that the position is updated only when the following two conditions are

met: (1) a better position is obtained; and (2) the probability $rand_3 < A_i(t)$ is satisfied. If the position of the ith bat is updated, then the corresponding loudness and emission rate $r_i(t+1)$ are replaced as follows:

$$A_i(t+1) = \alpha A_i(t), \tag{16}$$

$$r_i(t+1) = r(0) \cdot (1 - e^{-\gamma t}), \tag{17}$$

where $\alpha > 0$ and $\gamma > 0$ are two predefined parameters, and $A(0)$ and $r(0)$ are two initial values for the loudness and emission rate, respectively.

The pseudo code of the standard BA is listed in **Algorithm 1**:

Algorithm 1: *Standard bat algorithm*

```
Begin
    For each bat, initialize the position, velocity, and parameters;
    While (stop criterion is met)
      Randomly generate the frequency for each bat with Eq. (11)
      Update the velocity for each bat with Eq. (10);
      If rand_2 < r_i(t)
        Update the temp position for the corresponding bat with Eq. (12);
      Else
        Update the temp position for the corresponding bat with Eq. (13);
      End
      Evaluate its quality/fitness;
      Re-update the position for the corresponding bat with Eq. (15);
      If the position is updated
        Update the loudness and emission rate with Eq. (16) and Eq. (17),
        respectively;
      End
      Rank the bats and save the best position;
    End
    Output the best position;
End
```

3.2 Curve Strategy with BA

The local search of the BA is a perturbation strategy, and in the position update of generation $t+1$, if bat i is selected for the local search, it will produce the following perturbation near the global optimal value:

$$x_{ik}^{'}(t+1) = p_k(t) + \varepsilon_{ik} \overline{A}(t) \tag{18}$$

where ε_{ik} is a random vector that satisfies the uniform distribution between (-1 , 1), and $\overline{A}(t)$ is the average loudness of all bats at time t.0

This strategy considers using a perturbation parameter $\tau \cdot x_{\max}$ to replace the mean loudness $\overline{A}(t)$ and allows the parameter $\tau(t)$ to decrease linearly as the evolution algebra increases, thus modifying Eq. (18) to

$$\boldsymbol{x}_i'(t+1) = \boldsymbol{p}(t) + \boldsymbol{\varepsilon}_i \cdot \tau(t) \cdot x_{\max} \tag{19}$$

Because ε_i obeys the random vector that satisfies the uniform distribution in (-1, 1), the support set of the disturbed individual will be centered on the group Optimal $\boldsymbol{p}(t)$. The region of the radius $\tau(t)x_{\max}$ is searched, and the parameter $\tau(t)$ decreases linearly with an increase in the evolution algebra, after which the radius of the search region can be reduced gradually to improve the local search performance of the algorithm.

Eq. (20) describes the descending strategy of disturbance parameter $\tau(t)$:

$$\tau(t)=\tau_{\max}\cdot[1-(\frac{\tau_{\max}-\tau_{\min}}{\tau_{\max}\cdot(LG-1)}\cdot(t-1))^{k_1}]^{k_2} \tag{20}$$

When k_1=1 and k_2=1 , the formula is

$$\tau(t)=\tau_{\max}\cdot[1-(\frac{\tau_{\max}-\tau_{\min}}{\tau_{\max}\cdot(LG-1)}\cdot(t-1))] \tag{21}$$

In Eq. (20), t represents evolutionary generation, LG represents the largest evolution algebra, and the values of k_1 and k_2 determine different downward trends. **Fig. 2** shows the curves produced by different combinations of k_1 and k_2 when the evolutionary algebra t equals 3000, $\tau_{\max}$ takes 30%, and the lower bound $\tau_{\min}$ takes 0.01%.

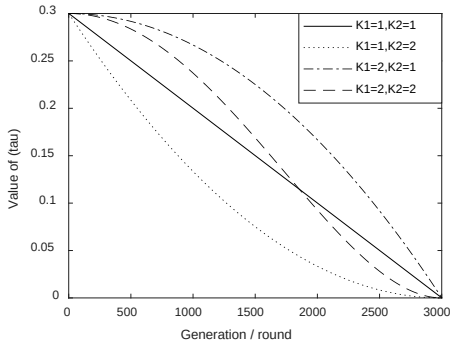

Fig. 2. Various curves of $\tau(t)$ variation.

Will different downward trends exert different effects on the algorithm? For example, if the decline in the early stage is slower and that in the later stage is slower, or slower in the early stage and faster in the later stage, can the performance of the algorithm be further improved? From this perspective, we discuss several declining curves of different parameters in this section.

Fig. 2 shows that the combination of k_1 and k_2 can roughly produce four curves: when $k_1=1, k_2=1$, it is a linear descending line; when $k_1=1$, $k_2=2,3,\ldots$, it is a concave curve; when $k_1=2,3,\ldots, k_2=1$, it is a convex curve; and when $k_1=2,3,\ldots$, $k_2=2,3,\ldots$, it is a front convex back concave curve. In addition to the linearly declining line, the remaining three curves are a few more representative typical curves introduced in this paper. In the subsequent argument selection, we discuss the value of $k_1=1$ and $k_2=1$. To demonstrate convenience, the algorithm is called BA with curve parameter (BA-CP). The process is illustrated in **Algorithm 2**.

The curve-decrease strategy algorithm is proposed based on the linear decline strategy. In the experimental section, we determine which curves perform better when $\tau \in [0.01\%, 30\%]$.

In the next part, we combine the curve strategy with FTBA and propose FTBA-TC. Then, we modify LEACH with the new algorithm.

Algorithm 2: *Bat algorithm with curve parameter*

Begin
For each bat, initialize $\boldsymbol{x}_i(0)$, $\boldsymbol{v}_i(0)$, $\boldsymbol{r}(0)$, $\tau_{\max}$, $\tau_{\min}$, k_1, k_2 and $fr_i(0)$;
While (stop criterion is met)
Calculate the fitness value of each bat, and remember that $\boldsymbol{p}(0)$ is the best position for group performance; set $t = 0$,
$f(\boldsymbol{p}(t)) = \min\{f(\boldsymbol{x}_i(0)) \mid i = 1, 2, ..., n\}$;
Update the velocity for each bat in the $t+1$ generation;
If $rand_1 < r_i(t)$
Update the new position $\boldsymbol{x}_i'(t+1)$ for corresponding bat with Eq. (12);
Else
Update the new position $\boldsymbol{x}_i'(t+1)$ for corresponding bat with Eq. (19) and Eq. (20);
End
Evaluate its quality/fitness for the new position of bat at $\boldsymbol{x}_i'(t+1)$;
If $f(\boldsymbol{x}_i'(t+1)) < f(\boldsymbol{x}_i'(t))$
Update the new position $\boldsymbol{x}_i'(t+1) = \boldsymbol{x}_i'(t+1)$, and update the $fr_i(t)$;
End
Re-update the position for the corresponding bat with $\boldsymbol{p}(t+1)$;
End
Output the best position $\boldsymbol{p}(t+1)$;
End

3.3 Proposed FTBA-TC-LEACH

FTBA was proposed by Cai to effectively improve the global search performance. In this part, we will combine the curve strategy with FTBA and the proposed fast BA with triangle flip and curve strategy (FTBA-TC).

In BA, when a better position is found, the position is updated to a more preferable one; when a better position is not found, the bats are suspended. Following this rule, Cui proposed a multiple triangle-flipping strategy to enhance the performance.

(1) BA with directing triangle-flipping strategy (BA-DTFS);
(2) BA with directing triangle-flipping strategy for all bats without conditions (BA-DTFS1);
(3) BA with random triangle-flipping strategy (BA-RTFS);
(4) BA with random triangle-flipping strategy for all bats without conditions (BA-RTFS1); and
(5) BA with hybrid triangle-flipping strategy (BA-HTFS).

Simulation results show that the BA-HTFS was superior, so we will use it as FTBA. In FTBA, the improved equations are shown below:

$$\boldsymbol{v}_i(t+1) = (\boldsymbol{x}_m(t) - \boldsymbol{x}_u(t)) \cdot fr_i(t) \tag{22}$$

$$\boldsymbol{v}_i(t+1) = (\boldsymbol{p}(t) - \boldsymbol{x}_m(t)) \cdot fr_i(t) \tag{23}$$

$$x_{\text{gk}}(t+1) = x_{\min} + (x_{\max} - x_{\min}) \cdot rand \tag{24}$$

In the eauation (22), the $\boldsymbol{x}_m(t)$ and $\boldsymbol{x}_u(t)$ are the locations of two randomly selected individuals in the population, the $fr_i(t)$ is the frequency of generation t, $\boldsymbol{v}_i(t+1)$ means the velocity of the next generation. The equation has 50% probability that the algorithm can improve performance, while the other half of the probability is used to improve the algorithm to jump out of the local extremum point. As for equation (23), $\boldsymbol{p}(t)$ is the the population previous best position, and other parameters are defined in the same way as equation (22). This equation is used to strengthen the local search capability in later stage. And equation (24), $rand$ is a random number that satisfies a uniform distribution between (0,1), $x_{\max}$ and $x_{\min}$ are the maximum and minimum values of the defined domain. $x_{gk}(t+1)$ indicates the location of the next generation of k dimensions in the g bat. The equation (24) can have a certain probability to jump out of the local optimal.

Algorithm 3 presents the process of FTBA-TC.

Algorithm 3: *Fast bat algorithm with triangle flip and curve strategy*

Begin
For each bat, initialize $\boldsymbol{x}_i(0)$, $\boldsymbol{v}_i(0)$, $\boldsymbol{r}(0)$, $\tau_{\max}$, $\tau_{\min}$, k_1, k_2 and $fr_i(0)$;
While (stop criterion is met)
Calculate the fitness value of each bat, and remember that $\boldsymbol{p}(0)$ is the best position for group performance; set $t = 0$,
$f(\boldsymbol{p}(t)) = \min\{f(\boldsymbol{x}_i(0)) \mid i = 1, 2, ..., n\}$;
If $t < 0.258 \cdot LG$
For each bat, randomly select $\boldsymbol{x}_m(t)$, $\boldsymbol{x}_u(t)$ and update the $t+1$ generation's velocity in Eq. (22).
Else
Update the $t+1$ generation's velocity in Eq. (23).
End
If $rand_1 < r_i(t)$
Update the new position $\boldsymbol{x}_i'(t+1)$ for corresponding bat with Eq. (12);
Else
Update the new position $\boldsymbol{x}_i'(t+1)$ for corresponding bat with Eq. (20);
End
For the bat in the best position for group history, let the position be randomly selected according to Eq. (24);
Evaluate its quality/fitness for new position of the bat at $\boldsymbol{x}_i'(t+1)$;
If $f(\boldsymbol{x}_i'(t+1)) < f(\boldsymbol{x}_i'(t))$
Update the new position $\boldsymbol{x}_i(t+1) = \boldsymbol{x}_i'(t+1)$ and update the $fr_i(t)$;
End
Re-update the position for the corresponding bat with $\boldsymbol{p}(t+1)$;
End
Output the best position $\boldsymbol{p}(t+1)$;
End

During selection, the temporary cluster-head is generated first, the average residual energy in clusters is calculated, and then the modified BA is used to find the optimal position in the cluster. If the temporary node's energy is more than the average energy, the node becomes the official head node and broadcasts the identity message; if the node energy is lower than the average energy, the algorithm waits for an official node. This process loops until all clusters are complete, and then the protocol enters the ready phase.

We replace the original selection algorithm with the modified BA in the protocol and use the intelligent algorithm to select the best position of the cluster-head node. The post-fusion protocol can find the cluster-head node accurately in the cluster; simulation results in the final section indicate effectiveness.

And **Fig. 3** shows the algorithm flow of the LEACH protocol combined with the intelligent optimization algorithm.

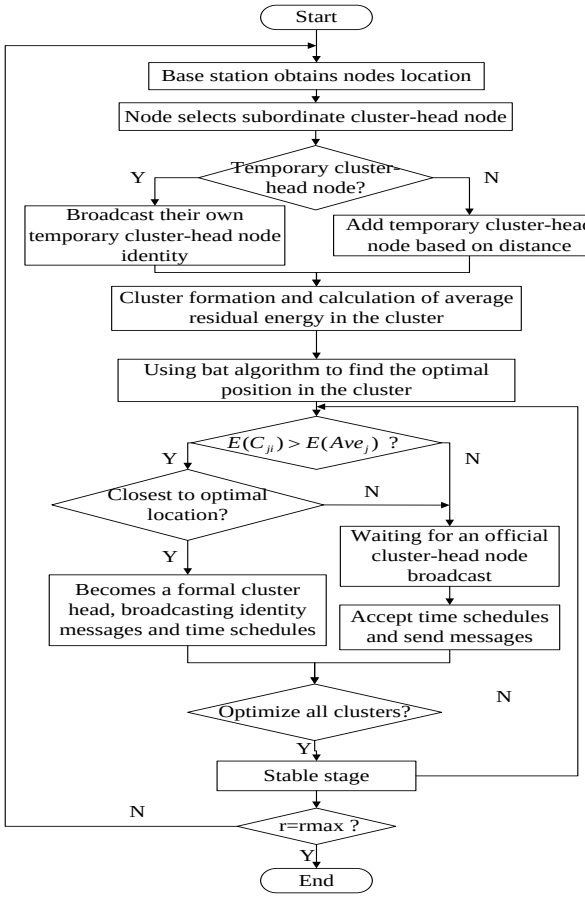

Fig. 3. LEACH protocol flow chart with intelligent optimization algorithm.

4. Simulation Results and Analysis

In this section, we verify the effectiveness of the proposed algorithm via the following experiments: (1) effectiveness of FTBA-TC on numerical optimization, and (2) effectiveness of FTBA-TC-LEACH. All experiments were implemented in MATLAB.

4.1 Tuning Performance

To verify the performance of FTBA-TC, we compared the algorithm with others to solve the CEC 2013 dataset [46]. The experimental index and parameter settings are listed in **Table 1**. Among them, the population size, dimensions, independent operating times, fitness value evaluation times, search domain, and other parameters were set according to the requirements of the CEC2013 test set. The nonparametric tests we adopted included two methods, the *Friedman* test and *Wilcoxon* test. The *Friedman* test is used to detect differences in treatments across multiple test attempts, and the *Wilcoxon* test detects whether this algorithm was significantly different from others.

Table 1. Parameter settings for bat algorithm.

Search domain	$[-100, 100]^D$
Frequency	[0.0, 5.0]
Initial $A_i(0)$	0.95
Initial $r_i(0)$	0.9
α	0.99
γ	0.9
Dimension D	30
Independent operation times	51
Fitness value evaluation times	300000
Population size	100

The experimental environment is as follows, where the test function set and parameters were identical. For convenient analysis, we labelled the concave curve as CP1, the convex curve as CP2, and the front convex back concave curve as CP3. k_1 and k_2 values are shown in **Table 2**.

(1) Experiment 1: Performance comparison of three curves.

First, consider the values of k_1 and k_2 to determine which curve performance is superior. **Table 2** shows the results of the *Friedman* test for these four settings; the performance of CP1 was best. The *Wilcoxon* test results in **Table 3** reveal a significant difference between CP1 and the other three settings; hence, the performance of the concave-type search curve was excellent.

Table 2. Initial value test results of k_1 and k_2.

Strategy	Curve type	(k_1, k_2)	Ranking
CP1	**Concave**	**(1 , 2)**	**1.59**
CP2	Convex	(2 , 1)	3.43
CP3	Pre-convex posterior concave	(2, 2)	2.38
LP6	Linear	(1 , 1)	2.61

Table 3. *Wilcoxon* test results.

CP1 vs.	*p*-value
CP2	**0.000**
CP3	**0.001**
LP6	**0.014**

(2) Experiment 2: Parameter selection of concave curve.

The above experiments show that the concave curve improves the average performance of the algorithm, and the concave curve is affected by parameter k_2. **Fig. 4** demonstrates the influence of four k_2 values on the concave curve; therefore, this experiment addresses how to select the value of k_2. As the different concave curves are mainly embodied in the selection of k_2, we define the following strategies according to the value of k_2: CP1-2, CP1-3, CP1-4, CP1-5, CP1-6, and CP1-7. The settings for these strategies appear in **Table 4**.

Table 4 presents the *Friedman* test of these strategies, ranked from small to large: CP1-4 < CP1-3 < CP1-5 < CP1-2 < CP1-6 < CP1-7. The performance of CP1-4 was therefore better. According to the *Friedman* test results in **Table 4**, the rankings show that CP1-4 was superior. As indicated in the righthand column, significant differences were found between CP1-4 and CP1-6 and CP1-7 in the *Wilcoxon* test. Results show that performance of the CP1-4 algorithm to be optimal when the $k_1 = 1$ and $k_2 = 4$.

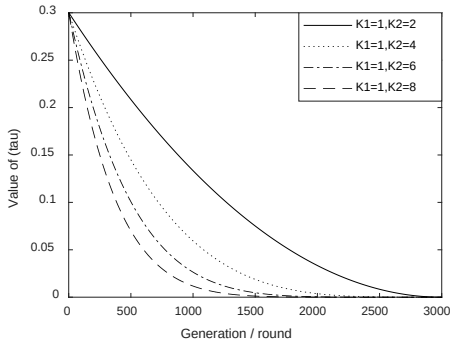

Fig. 4. Different shapes of concave curves.

Table 4. Different combinations and *Friedman* and *Wilcoxon* tests of k_1 and k_2.

Strategies	(k_1 , k_2)	*Friedman ranking*	*p*-value of *Wilcoxon*
CP1_2	(1 , 2)	3.34	0.872
CP1_3	(1 , 3)	3.18	0.376
CP1_4	**(1 , 4)**	**2.71**	-
CP1_5	(1 , 5)	3.21	0.248
CP1_6	(1 , 6)	3.96	**0.001**
CP1_7	(1 , 7)	4.59	**0.000**

4.3 Effectiveness of FTBA-TC on Numerical Optimization

The experiments were conducted in two parts: first, we compared the performance of the following three algorithms to verify whether FTBA-TC improved FTBA performance.

(1) For FTBA, we set $Threshold = 0.258$;
(2) for BA with curve strategy (CP1-4), we set $k_1 = 1$ and $k_2 = 4$; and
(3) FTBA-TC.

Table 5 shows the average error of these algorithms; w/t/l indicates that the performance of FTBA-TC was better. **Table 6** presents the Friedman test for these algorithms, ranked as FTBA-TC < CP1-4 < FTBA, confirming that the FTBA-TC performance of these three algorithms was superior. The *Wilcoxon* test results in **Table 7** indicate significant differences between FTBA-TC and the other two algorithms.

Table 5. 30-dimension mean comparison results.

Function	FTBA	CP1-4	FTBA-TC
F1	9.63E-05	6.51E-10	**1.07E-10**
F2	**4.22E+04**	4.04E+06	7.03E+04
F3	**7.75E+06**	2.10E+08	1.26E+07
F4	**8.90E-02**	1.63E+04	1.43E-01
F5	**1.11E-03**	7.44E-03	1.84E-03
F6	**6.72E+00**	4.27E+01	3.05E+01
F7	1.06E+02	7.24E+01	**3.05E+01**
F8	**2.09E+01**	**2.09E+01**	**2.09E+01**
F9	2.98E+01	2.02E+01	**1.56E+01**
F10	7.79E-02	4.00E-02	4.94E-02
F11	3.45E+02	**1.35E+02**	6.81E+01
F12	3.52E+02	1.31E+02	**6.41E+01**
F13	3.58E+02	2.19E+02	**1.26E+02**
F14	4.27E+03	3.43E+03	**2.95E+03**
F15	4.44E+03	3.32E+03	**3.09E+03**
F16	3.50E-01	3.66E-01	**1.78E-01**
F17	1.72E+02	1.66E+02	**9.09E+01**
F18	1.41E+02	1.54E+02	**9.05E+01**
F19	7.14E+00	6.29E+00	**3.99E+00**
F20	1.26E+01	1.28E+01	**1.04E+01**
F21	3.33E+02	**3.09E+02**	3.17E+02
F22	5.49E+03	3.97E+03	**3.19E+03**
F23	5.55E+03	3.95E+03	**3.25E+03**
F24	2.95E+02	2.56E+02	**2.32E+02**
F25	3.27E+02	2.88E+02	**2.71E+02**
F26	2.04E+02	**2.00E+02**	**2.00E+02**
F27	1.13E+03	8.67E+02	**7.06E+02**
F28	2.25E+03	4.09E+02	**3.53E+02**
w/t/l	22/1/5	24/2/2	

Table 6. *Friedman* test results.

Algorithms	Ranking
FTBA	2.50
CP1-4	2.20
FTBA-TC	**1.30**

Table 7. *Wilcoxon* test results.

FTBA-TC vs.	*p*-value
FTBA	**0.004**
CP1-4	**0.000**

Second, to further test the performance of the FTBA-TC algorithm, we compared it to the following four algorithms:

(1) PSO [47];

(2) PSO with time-varying accelerator coefficient (TVAC) [48];

(3) CS [49];

(4) Multi-strategy ensemble artificial bee colony algorithm (MEABC) [34].

Table 8 lists the average error in the CEC2013 test set for these five algorithms. From the w/t/l perspective, the deviation of seven functions in PSO was less than that of FTBA-TC, and the deviation of the other 19 functions was greater than that of FTBA-TC. TVAC included 20 functions whose deviations were greater than FTBA-TC; CS had 22 functions worse than FTBA-TC; and even for MEABC, 15 functions of deviations were inferior to FTBA-TC. It can be seen from table that many test functions are one or more orders of magnitude better than other algorithms. In conclusion, the average performance of FTBA-TC was superior to the other four algorithms.

Table 9 presents the *Friedman* test of five algorithms, ranked as FTBA-TC < MEABC < TVAC < CS < PSO, suggesting that the average performance of the FTBA-TC algorithm was superior to others. The average performance of PSO algorithm was worse than the other algorithms; the *Wilcoxon* test results in **Table 10** highlight significant differences between FTBA-TC and the other algorithms except MEABC.

Table 8. 30-dimension mean comparison results.

Function	PSO	TVAC	CS	MEABC	FTBA-TC
F1	0.00E+00	0.00E+00	3.90E-03	0.00E+00	**1.07E-10**
F2	1.86E+07	1.01E+07	**3.21E+02**	1.23E+06	7.03E+04
F3	4.77E+09	3.16E+08	3.66E+08	1.40E+08	**1.26E+07**
F4	1.95E+04	2.17E+03	2.17E+00	8.35E+04	**1.43E-01**
F5	**5.73E-06**	0.00E+00	4.72E-02	0.00E+00	1.84E-03
F6	2.50E+02	1.72E+02	1.50E+01	**1.01E+01**	3.05E+01
F7	9.98E+01	7.98E+01	8.69E+01	9.23E+01	**3.05E+01**
F8	**2.09E+01**	**2.09E+01**	2.10E+01	**2.09E+01**	**2.09E+01**
F9	2.83E+01	2.71E+01	3.09E+01	2.88E+01	**1.56E+01**
F10	2.50E+01	1.29E+01	1.06E-01	5.57E+00	**4.94E-02**

F11	3.62E+01	1.99E+01	**1.20E+02**	0.00E+00	6.81E+01
F12	1.23E+02	1.14E+02	1.80E+02	2.07E+02	**6.41E+01**
F13	2.14E+02	1.88E+02	2.20E+02	2.29E+02	**1.26E+02**
F14	9.04E+02	9.39E+02	2.72E+03	**1.37E+01**	2.95E+03
F15	6.92E+03	3.74E+03	4.08E+03	**3.41E+02**	3.09E+03
F16	2.35E+00	1.91E+00	2.05E+00	1.44E+00	**1.78E-01**
F17	6.86E+01	4.67E+01	1.84E+02	**3.04E+01**	9.09E+01
F18	2.31E+02	1.02E+02	2.03E+02	1.80E+02	**9.05E+01**
F19	1.61E+01	1.07E+01	1.05E+01	**3.94E-01**	3.99E+00
F20	1.29E+01	1.19E+01	1.38E+01	1.56E+01	**1.04E+01**
F21	3.34E+02	3.69E+02	2.13E+02	**2.10E+02**	3.17E+02
F22	7.26E+02	8.21E+02	3.33E+03	**1.78E+01**	3.19E+03
F23	7.41E+03	4.41E+03	5.07E+03	5.16E+03	**3.25E+03**
F24	2.82E+02	2.77E+02	2.70E+02	2.81E+02	**2.32E+02**
F25	3.26E+02	3.26E+02	3.18E+02	2.74E+02	**2.71E+02**
F26	**2.00E+02**	**2.00E+02**	**2.00E+02**	2.01E+02	**2.00E+02**
F27	9.35E+02	8.74E+02	1.01E+03	**4.02E+02**	7.06E+02
F28	3.21E+02	3.98E+02	3.33E+02	**3.00E+02**	3.53E+02
w/t/l	19/2/7	20/2/6	22/1/5	15/1/12	

Table 9. *Friedman* test results.

Algorithm	Rankings
PSO	3.77
TVAC	2.89
CS	3.59
MEABC	2.57
FTBA-TC	**2.18**

Table 10. *Wilcoxon* test results.

FTBA-TC vs.	*p*-value
PSO	**0.009**
TVAC	**0.007**
CS	**0.014**
MEABC	0.581

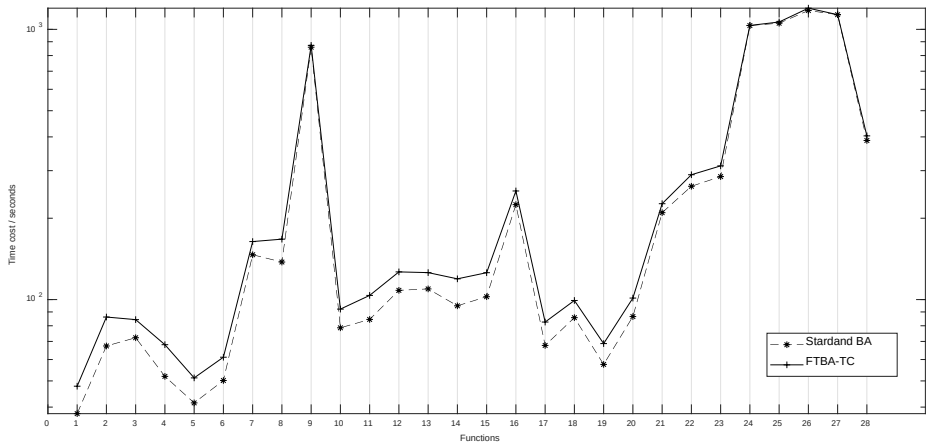

Fig. 5. The time cost of standard BA and FTBA-TC.

As can be seen from the above **Fig. 5**, the improved BA consumes a little more time than the standard BA, but the accuracy is higher than the original BA and works better, so this improvement is worthwhile.

4.4 Effectiveness of FTBA-TC-LEACH

To verify our improved LEACH protocol, we used the following algorithms for comparison:

(1) Standard LEACH protocol;

(2) LEACH protocol combined with standard BA (SBA-LEACH); and

(3) LEACH protocol combined with fast BA with triangle flip and curve strategy (FTBA-TC-LEACH).

Table 11. Parameter setting of simulation experiment.

Parameter	Value
Number of nodes	100
Initial energy of node	0.5J
Maximum number of cycles	2000
data pack	4000bits
Energy consumption for sending and receiving unit data	50nJ/bit
Fusion unit data consumes energy	5nJ/bit
Node becomes cluster-head node probability	0.05

The simulation environment was $100m \times 100m$, in which sensor nodes were randomly distributed and the base station was at (50 , 50). Assume the base station has unlimited energy and does not consider communication conflicts or communication delay between nodes. Specific simulation parameters are shown in **Table 11**.

Fig. 6 presents a comparison of the LEACH protocol, SBA-LEACH protocol, and FTBA-TC-LEACH protocol life cycle based on survival node numbers. Starting from the 800^{th} generation, the survival point of the LEACH protocol began to decline sharply and was lowest, occurring mainly in the 800^{th}–1300^{th} generations. After 1300 generations, the survival point declined by less than the SBA-LEACH protocol and FTBA-TC-LEACH protocol. Since the 1800s, the degree of decline increased dramatically; hence, the LEACH protocol was

deemed inferior to the other two protocols in terms of survival points. Compared with the SBA-LEACH protocol and FTBA-TC-LEACH protocol, the difference in the survival point of the FTBA-TC-LEACH protocol was greater than the SBA-LEACH protocol starting from the 1400th generation.

Fig. 7 indicates that the residual energy of the LEACH protocol, SBA-LEACH protocol, and FTBA-TC-LEACH protocol demonstrated little difference before 1400 generations, but as the algebra continued to increase, the energy of the LEACH protocol was consumed rapidly. The rate of decline was much faster than in the SBA-LEACH protocol and FTBA-TC-LEACH protocol; the same was true for the SBA-LEACH protocol. As can be seen from **Table 12**, after 2000 generations, the number of nodes remaining in the FTBA-TC-LEACH protocol was highest, nearly twice that of the LEACH protocol and greater than the SBA-LEACH by approximately 37.5%. The total residual energy of the FTBA-TC-LEACH protocol was 1.63 times that of the LEACH protocol and 1.41 times that of the SBA-LEACH protocol; therefore, the performance of the FTBA-TC-LEACH protocol was superior.

Table 12. Comparison results of three protocols.

Algorithm	LEACH	SBA-LEACH	FTBA-TC-LEACH
Number of surviving nodes	6	8	11
Total remaining energy of the node	1.0350	1.2013	1.6879

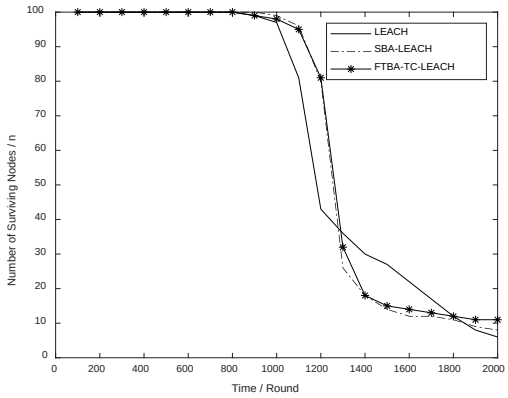

Fig. 6. Comparison of life cycle.

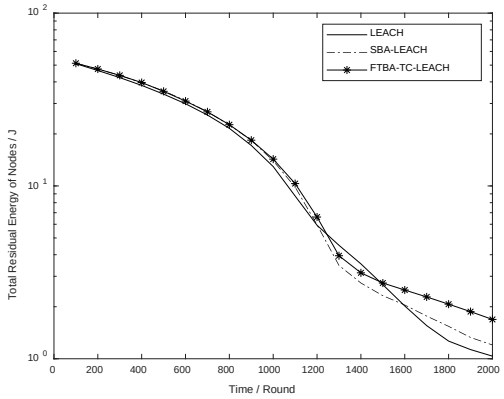

Fig. 7. Comparison of node residual energy.

5. Conclusion and Future Work

LEACH is a well-known algorithm used in WSNs to reduce system energy costs. In this paper, we improved the cluster-head node selection in LEACH. Because the cluster-head node is selected randomly, the cluster-head node may be far from the base station and the remaining energy may be poorly distributed in the system, leading to premature node death. To solve this problem, we modified the BA to optimize cluster-head node selection and proposed a curve strategy with FTBA (FTBA-TC). FTBA-TC enhanced the local search ability based on FTBA and improved the global and local search capacity of BA. Three different curve shapes and six different parameter combinations were tested using experiments. The simulation of the

FTBA-TC was superior to the otsher algorithms, namely PSO, TVAC, CS, and MEABC. Then, the algorithm was applied to the LEACH protocol. Experiments compared standard LEACH, SBA-LEACH, and FTBA-TC-LEACH, the latter of which was most effective.

Using BA to solve pertinent WSN problems is important. Node localization is a key technology in WSNs, and least square algorithm has low localization precision. WANG Zhanbei [50] used the BA to solve this problem. To solve the node location error in WSNs, SHANG Junna [51] proposed a new multi-agent bat algorithm with favorable local searching ability. In our future work, we will use a modified BA for relevant problem optimization.

Acknowledgement

This work is supported by the National Natural Science Foundation of China under Grant No.61806138, No.U1636220 and No.61663028, Natural Science Foundation of Shanxi Province under Grant No.201601D011045, PhD Research Startup Foundation of Taiyuan University of Science and Technology under Grant No.20182002.

References

[1] M. A. Zu-Chang, Y. N. Sun, and T. Mei, "Survey on wireless sensors network," *Journal of China Institute of Communications*, vol. 20, no. 7, pp. 1737-1880, 2004. Article (CrossRef Link).

[2] V. Potdar, A. Sharif, and E. Chang, "Wireless Sensor Networks: A Survey," in *Proc. of International Conference on Advanced Information Networking and Applications Workshops*, pp. 636-641, 2009. Article (CrossRef Link).

[3] F. Y. Ren et al., "Wireless Sensor Networks," *Journal of Software*, vol. 14, no. 7, p. 1282-1291, 2003. Article (CrossRef Link).

[4] Z. Cui, F. Xue, X. Cai, Y. Cao, G. G. Wang, and J. Chen, "Detection of Malicious Code Variants Based on Deep Learning," *IEEE Transactions on Industrial Informatics*, vol. 14, no. 7, pp. 3187-3196, 2018. Article (CrossRef Link).

[5] K. Akkaya and M. Younis, "A survey on routing protocols for wireless sensor networks," *Ad Hoc Networks*, vol. 3, no. 3, pp. 325-349, 2005. Article (CrossRef Link).

[6] S. Sharma and M. Kumar, "LEACH PROTOCOL: A Survey," *International Journal of Computer Science & Communication Networks*, vol. 5(4), pp. 228-232, 2015. Article (CrossRef Link).

[7] W. R. Heinzelman, A. Chandrakasan, and H. Balakrishnan, "Energy-efficient communication protocol for wireless microsensor networks," in *Proc. of Hawaii International Conference on System Sciences*, vol. 2, pp. 10, 2000. Article (CrossRef Link).

[8] A. Manjeshwar and D. P. Agrawal, "TEEN: A Routing Protocol for Enhanced Efficiency in Wireless Sensor Networks," in *Proc. of Parallel and Distributed Processing Symposium., Proceedings International*, p. 189, 2002.

[9] S. Lindsey and C. S. Raghavendra, "PEGASIS: Power-efficient gathering in sensor information systems," in *Proc. of Aerospace Conference Proceedings*, vol. 3, pp. 3-1125-3-1130, 2003. Article (CrossRef Link).

[10] H. U. Gang, D. M. Xie, and W. U. Yuan-Zhong, "Research and Improvement of LEACH for Wireless Sensor Networks," *Chinese Journal of Sensors & Actuators*, vol. 20, no. 6, pp. 1391-1396, 2007. Article (CrossRef Link).

[11] Y. Hu and J. Wang, "Analysis of LEACH Protocol Based on Ant Algorithm," *Advanced Materials Research*, vol. 287-290, no. 5, pp. 2830-2833, 2011. Article (CrossRef Link).

[12] Z. Wei-hua, L. La-yuan, Z. Liu-min, and W. Xuan-zheng, "Energy Consumption Balance Improvement of LEACH of WSN," *Chinese Journal of Sensors & Actuators*, vol. 21, no. 11, pp. 1918-1922, 2008.

[13] Z. Cui, Y. Cao, X. Cai, J. Cai, and J. Chen, "Optimal LEACH protocol with modified bat algorithm for big data sensing systems in Internet of Things," *Journal of Parallel & Distributed Computing*, 2018. Article (CrossRef Link).
[14] L. I. Yan, X. H. Zhang, and L. I. Yan-Zhong, "Algorithm of cluster head multi-hops based on LEACH," *Computer Engineering & Design*, vol. 17, pp. 100-102, 2007. Article (CrossRef Link).
[15] W. W. Niu and T. G. Gao, "Improvement algorithm for simulated annealing in LEACH-C protocol," *Computer Engineering & Design*, vol. 32, no. 6, pp. 1869-290, 2011. Article (CrossRef Link).
[16] W. Lin and J. Pan, "ANT-LEACH:LEACH routing protocol integrated energy optimization by ant colony optimization in wireless sensor network," *Journal of Computer Applications*, vol. 31, no. 11, pp. 2891-2894, 2011. Article (CrossRef Link).
[17] D. X. Han, R. H. Zhang, and D. H. Liu, "PSO-based Double Cluster-heads Clustering Algorithm for Wireless Sensor Network," *Computer Engineering*, vol. 36, no. 10, pp. 100-102, 2010. Article (CrossRef Link).
[18] Wang et al., "Diversity enhanced particle swarm optimization with neighborhood search," *Information Sciences*, vol. 223, no. 2, pp. 119-135, 2013. Article (CrossRef Link).
[19] J. Jiang, "An improved LEACH protocol in wireless sensor networks," *Journal of Soochow University*, vol. 01, pp. 4-8, 2011. Article (CrossRef Link).
[20] S. Sadeghiram, "Bacterial foraging optimisation algorithm, particle swarm optimisation and genetic algorithm: a comparative study," *International Journal of Bio-Inspired Computation (IJBIC)*, vol. 10, no. 4, pp. 275-282, Nov 2017. Article (CrossRef Link).
[21] W. A. K. R. B. S. L. J. Venstrom, "A parameter estimation method for stiff ordinary differential equations using particle swarm optimisation," *International Journal of Computing Science and Mathematics (IJCSM)*, vol. 9, no. 5, pp. 419-432, Sep 2018. Article (CrossRef Link).
[22] S. Okdem and D. Karaboga, "Routing in Wireless Sensor Networks Using Ant Colony Optimization," in *Proc. of Nasa/esa Conference on Adaptive Hardware and Systems, ed*, pp. 401-404, 2006. Article (CrossRef Link).
[23] P. Stodola and J. Mazal, "Applying the ant colony optimisation algorithm to the capacitated multi-depot vehicle routing problem," International Journal of Bio-Inspired Computation (IJBIC), vol. 8, no. 4, pp. 228-233, 2016. Article (CrossRef Link).
[24] M. Z. Y. Zhang, "Ant colonial-based approach for the minimal full trie problem," *International Journal of Bio-Inspired Computation (IJBIC)*, vol. 9, no. 4, pp. pp.235 - 239, May 2017. Article (CrossRef Link).
[25] J. L. L. K. G. Y. T. Zhang, "Ant colony optimisation for the routing problem in the constellation network with node satellite constraint," *International Journal of Bio-Inspired Computation (IJBIC)*, vol. 10, no. 4, pp. 267 - 274, Nov 2017. Article (CrossRef Link).
[26] A. G.-P. J. D. S. D. Camacho, "Solving strategy board games using a CSP-based ACO approach," *International Journal of Bio-Inspired Computation (IJBIC)*, vol. 10, no. 2, pp. 136 - 144, july 2017. Article (CrossRef Link).
[27] R. H. C. M. X. J. Q. X. L. Qi, "Optimisation of dangerous goods transport based on the improved ant colony algorithm," *International Journal of Computing Science and Mathematics (IJCSM)*, vol. 8, no. 3, pp. 210-217, Jul 2017. Article (CrossRef Link).
[28] Z. Cui, B. Sun, G. Wang, Y. Xue, and J. Chen, "A novel oriented cuckoo search algorithm to improve DV-Hop performance for cyber-physical systems," *Journal of Parallel & Distributed Computing*, vol. 103, pp. 42-52, 2017. Article (CrossRef Link).
[29] X. S. Yang and S. Deb, "Engineering Optimisation by Cuckoo Search," *International Journal of Mathematical Modelling & Numerical Optimisation*, vol. 1, no. 4, pp. 330-343, 2010. Article (CrossRef Link).
[30] Z. Cui, B. Sun, G. Wang, Y. Xue, and J. Chen, "A novel oriented cuckoo search algorithm to improve DV-Hop performance for cyber–physical systems," *Journal of Parallel and Distributed Computing*, vol. 103, pp. 42-52, 2017. Article (CrossRef Link).
[31] M. Zhang, H. Wang, Z. Cui, and J. Chen, "Hybrid multi-objective cuckoo search with dynamical local search," *Memetic Computing*, no. 10, no. 2, pp. 199-208, 2018. Article (CrossRef Link).

[32] W.-H. Y. J.-R. L. Y. Zhang, "A new local-enhanced cuckoo search algorithm," *International Journal of Computing Science and Mathematics (IJCSM)*, vol. 8, no. 2, pp. 175-182, Apr 2017. Article (CrossRef Link).
[33] D. Karaboga, "An Idea Based on Honey Bee Swarm for Numerical Optimization," *TECHNICAL REPORT-TR06*, 2005. Article (CrossRef Link).
[34] H. Wang, Z. Wu, S. Rahnamayan, H. Sun, Y. Liu, and J. S. Pan, "Multi-strategy ensemble artificial bee colony algorithm," *Information Sciences*, vol. 279, pp. 587-603, 2014. Article (CrossRef Link).
[35] N. S. J. M.-S. A. G. Abro, "Robust variant of artificial bee colony (JA-ABC4b) algorithm," *International Journal of Bio-Inspired Computation (IJBIC)*, vol. 10, no. 2, pp. 99-108, July 2017. Article (CrossRef Link).
[36] S. Z. Y. Z. Z. Z. D. Z. H. Zhang, "Comparative analysis of selection schemes used in artificial bee colony algorithm," *International Journal of Computing Science and Mathematics (IJCSM)*, vol. 8, no. 3, pp. 218-227, Jul 2017. Article (CrossRef Link).
[37] X. Y. Y. M. Z. Liu, "An improved artificial bee colony algorithm for solving parameter identification problems," *International Journal of Computing Science and Mathematics (IJCSM)*, vol. 8, no. 6, pp. 570-579, Dec 2017. Article (CrossRef Link).
[38] M. D. Y. B. Y. Wang, "A membrane-inspired quantum bee colony optimisation based on simulated quantum gate and its applications," *International Journal of Computing Science and Mathematics (IJCSM)*, vol. 8, no. 5, pp. 465-474, Nov 2017. Article (CrossRef Link).
[39] X. S. Yang, "A New Metaheuristic Bat-Inspired Algorithm," *Nature Inspired Cooperative Strategies for Optimization (NICSO 2010)*, vol. 284, pp. 65-74, 2010. Article (CrossRef Link).
[40] X. Cai, X. Z. Gao, and Y. Xue, "Improved bat algorithm with optimal forage strategy and random disturbance strategy," *International Journal of Bio-Inspired Computation (IJBIC)*, vol. 8, no. 4, 2016. Article (CrossRef Link).
[41] F. L. Zhihua Cui, Wensheng Zhang, "Bat Algorithm with Principal Component Analysis," *International Journal of Machine Learning and Cybernetics*, 2018.
[42] N. M. Y. M. O. Tokhi, "A modified bats echolocation-based algorithm for solving constrained optimisation problems," *International Journal of Bio-Inspired Computation (IJBIC)*, vol. 10, no. 1, pp. 12 - 23, July 2017. Article (CrossRef Link).
[43] L. Li and Y. Zhou, "A novel complex-valued bat algorithm," *Neural Computing & Applications*, vol. 25, no. 6, pp. 1369-1381, 2014. Article (CrossRef Link).
[44] X. Cai, H. Wang, Z. Cui, J. Cai, Y. Xue, and L. Wang, "Bat algorithm with triangle-flipping strategy for numerical optimization," *International Journal of Machine Learning & Cybernetics*, vol. 9, no. 2, pp. 199-215, 2017. Article (CrossRef Link).
[45] L. I. Zhi-Yong, M. A. Liang, H. Z. Zhang, and S. O. Management, "Application of Bat Algorithm in Multi-Objective and Multi-Choice Knapsack Problem," *Computer Simulation*, vol. 30, no. 10, pp. 350-353, 2013. Article (CrossRef Link).
[46] J. J. Liang, B. Y. Qu, and P. N. Suganthan, "Problem definitions and evaluation criteria for the CEC 2013 special session and competition on real-parameter optimization," *Technical Report 201212 Computational Intelligence Laboratory, Zhengzhou University, Zhengzhou China And Technical Report, Nanyang Technological University, Singapore*, January, 2012. Article (CrossRef Link).
[47] Y. E. R. C. Shi, "A modified particle swarm optimizer," in *Proc. of 1998 IEEE International Conference on Evolutionary Computation Proceedings. IEEE World Congress on Computational Intelligence(no. 98TH8360)*, pp. 69-73, 4-9 May 1998. Article (CrossRef Link).
[48] A. Ratnaweera, S. K. Halgamuge, and H. C. Watson, "Self-organizing hierarchical particle swarm optimizer with time-varying acceleration coefficients," in *Proc, of IEEE Transactions on Evolutionary Computation*, vol. 8, no. 3, pp. 240-255, 2004. Article (CrossRef Link).
[49] X. S. Yang and S. Deb, "Cuckoo Search via Levy Flights," *2009 World Congress on Nature & Biologically Inspired Computing*, pp. 210 - 214, January 2010. Article (CrossRef Link).
[50] Z. Wang, "Node localization of wireless sensor networks based on bat algorithm," *Computer Engineering & Applications*, vol. 11, pp. 90-94, 2014. Article (CrossRef Link).

[51] J. Shang, C. Liu, K. Yue, and L. I. Lin, "The Multi-Agent Bat Algorithm Applied to Wireless Sensor Networks," *Chinese Journal of Sensors and Actuators*, vol. 9, pp. 1418-1424, 2015. Article (CrossRef Link).

Xingjuan Cai received her PhD in Control Science and Engineering from Tongji University, China in 2017, and BSc in Computer Science from Taiyuan Heavy Machinery Institute in 2008. Her research interest includes computational intelligence, stochastic algorithm, and combinatorial optimization.

Youqiang Sun is an academic master of Computer Science and Technology from Taiyuan University of Science and Technology, China. His research interest includes computational intelligence and deep learning.

Zhihua Cui is a Professor of Computer Science and Technology, and Director of Complex System and Computational Intelligence Laboratory at Taiyuan University of Science and Technology, China. He is a member of IEEE and ACM, and a senior member of China Computer Federation (CCF) and member of Chinese Association of Artificial Intelligence (CAAI). He received his Ph.D. in System Engineering from Xi'an Jiaotong University, China in 2008, and B.Sc. in Computer Science from Taiyuan Heavy Machinery Institute in 2003. He is the founding Editor-in-Chief of International Journal of Bio-inspired Computation. His research interest includes computational intelligence, stochastic algorithm, and combinatorial optimization. He has published over 60 peer reviewed journal papers, 60 peer reviewed full conference papers, and five books in computational intelligence.

Wensheng Zhang received the Ph.D. degree in Pattern Recognition and Intelligent Systems from the Institute of Automation, Chinese Academy of Sciences (CAS), in 2000. He joined the Institute of Software, CAS, in 2001. He is a Professor of Machine Learning and Data Mining and the Director of Research and Development Department, Institute of Automation, CAS. His research interests include computer vision, pattern recognition, artificial intelligence and computer human interaction.

Jinjun Chen received the PhD degree in computer science and software engineering from Swinburne University of Technology, Australia. He is an associate professor in the Faculty of Engineering and IT, University of Technology Sydney (UTS), Australia. He is the director of Lab of Cloud Computing and Distributed Systems. His research interests include cloud computing, big data, workflow management, privacy and security, and related various research topics. He is a senior member of the IEEE.