Hindawi Publishing Corporation
International Journal of Distributed Sensor Networks
Volume 2015, Article ID 729680, 10 pages
http://dx.doi.org/10.1155/2015/729680

Research Article

A Balanced Power Consumption Algorithm Based on Enhanced Parallel Cat Swarm Optimization for Wireless Sensor Network

Lingping Kong,[1] Jeng-Shyang Pan,[2] Pei-Wei Tsai,[2] Snasel Vaclav,[3] and Jiun-Huei Ho[4]

[1] *Innovative Information Industry Research Center, Shenzhen Graduate School, Harbin Institute of Technology, Shenzhen 518055, China*
[2] *College of Information Science and Engineering, Fujian University of Technology, Fuzhou 350118, China*
[3] *Department of Computer Science, VSB Technical University of Ostrava, 70833 Ostrava, Czech Republic*
[4] *Department of Computer Science and Information Engineering, Cheng Shiu University, Kaohsiung 83347, Taiwan*

Correspondence should be addressed to Jeng-Shyang Pan; jengshyangpan@gmail.com

Received 24 October 2014; Revised 26 January 2015; Accepted 5 February 2015

Academic Editor: Qiangfu Zhao

The wireless sensor network (WSN) is composed of a set of sensor nodes. It is deemed suitable for deploying with large-scale in the environment for variety of applications. Recent advances in WSN have led to many new protocols specifically for reducing the power consumption of sensor nodes. A new scheme for predetermining the optimized routing path is proposed based on the enhanced parallel cat swarm optimization (EPCSO) in this paper. This is the first leading precedent that the EPCSO is employed to provide the routing scheme for the WSN. The experimental result indicates that the EPCSO is capable of generating a set of the predetermined paths and of smelting the balanced path for every sensor node to forward the interested packages. In addition, a scheme for deploying the sensor nodes based on their payload and the distance to the sink node is presented to extend the life cycle of the WSN. A simulation is given and the results obtained by the EPCSO are compared with the AODV, the LD method based on ACO, and the LD method based on CSO. The simulation results indicate that our proposed method reduces more than 35% power consumption on average.

1. Introduction

The wireless sensor network (WSN) is a popular research field in computer science and telecommunications. It is extensively used in a variety of applications such as the environmental observation [1], the air pollution monitoring, the natural disaster prevention, and the healthcare. The WSN is composed of a few to hundreds or even thousands of sensor node modules; sensor nodes are capable of cooperating with others by passing packages through the network to the sink node, which is regarded as the data collection and control center. The price of a sensor node should be inexpensive. It implies that the battery, which supplies the power for the sensor node, is with a limited capacity. In many real-world applications, the sensor nodes are stochastic deployed by airplanes, vehicles, or other transportations. The distance and the density between sensor nodes are not identical. Without properly designing the deployment density, some of the sensor nodes would take higher payload of forwarding packages. This kind of situation is very common to be observed on the inner layer nodes located near to the sink node. Since the inner layer nodes have higher payload than the outer layer nodes, the life cycle of the inner layer nodes is shorter than those deployed in the outer layer. Once the inner layer nodes are out of battery, the whole WSN is paralyzed because the packages cannot be forwarded to the sink node. Therefore, a scheme for extending the life cycle of the whole WSN is employed at phase of the sensor node deployment. In addition, the connectivity and the coverage of the sensor nodes also affect the performance of the whole WSN, directly. For example, when using the WSN to monitor the environment in an area with natural disasters, the lack of full coverage is not tolerable. An effective WSN must provide the best coverage to meet the need of the user's requirement. For deploying structural network, each node plays the same role with the fixed and equalized sensing and transmitting range. These sensor nodes work

collaboratively for both sensing and broadcasting packages and transmit the interested packages to the relay nodes. The relay nodes will further forward the package to the sink node. Each sensor node can be treated as the relay node to its neighborhoods. Moreover, the sensor nodes are usually small and inexpensive. Considering the deployment method and the environment of using the WSN, the battery module on the sensor node is generally neither changeable nor rechargeable. The life cycle of the whole WSN can be extended by properly designing the deployment of the sensor nodes and employing the suitable routing algorithm. Hence, how to design an efficient data forwarding path and to maximally extend the life cycle of the WSN become the principal issues. Answering to these needs, enhanced parallel cat swarm optimization (EPCSO) [2–5] is modified partially and is employed to produce the balanced paths for the whole WSN. This is the first application that utilizes EPCSO in the routing design for WSN.

In this paper, we propose a strategy for deploying the sensor nodes by considering the coverage of WSN base on the minimum-number-node theory. Furthermore, we utilize EPCSO to design a routing algorithm for providing the balanced routing paths. Our design balances the power consumption between the forwarding distance and the energy saving for all nodes in the whole WSN. The rest of the paper is composed as follows: the related works on the minimum-number-node theory and EPCSO algorithm are briefly reviewed in Section 2; our proposed sensor node deployment strategy is presented in Section 3; our proposed method by using EPCSO to produce the balanced routing paths for the WSN is explained in detail in Section 4; the simulation result is given and is compared with the AODV method, the LD methods with both ACO and CSO in Section 5; finally, the conclusion is given in Section 6.

2. Related Works

2.1. Minimum-Number-Node Theory. In general, a WSN in the real-world application contains hundreds or even thousands of sensor nodes scattered in the network field. Keeping the WSN in a high connectivity status with full sensing coverage in a certain region is a difficult problem. The full sensing coverage can be secured by deploying massive sensor nodes into the field. However, deploying too many redundant sensor nodes into the field is very wasteful. Thus, the issue on the minimum requirement of the sensor node number has been discussed in many exiting literatures [8, 9]. In 2006, Liu et al. present the definition of coverage intensity [9] for a specific point (C_p) and network coverage intensity (C_n) as follows:

$$C_p = \frac{T_c}{T_a}, \tag{1}$$

where C_p is the coverage intensity for a given point p in an environment, T_a stands for any given long time period, and T_c is the total time during T_a when point p is covered by at least one active sensor. At the same time, sensors are independently and uniformly deployed in the environment. Hence, the expectation of C_p is equal and could be used to evaluate the coverage quality of the whole network. Thus, the definition of the network coverage intensity C_n is equal to the expectation of C_p [9] and is given as follows:

$$C_n = E\left|C_p\right| = 1 - \left(1 - \frac{q}{k}\right)^n, \quad \text{where } q = \frac{r}{a}, \tag{2}$$

where q is the probability that each sensor node covers a given point and k denotes the number of subset. The minimum-number-node theory is a corollary from C_n. For C_n, if we predefine the subset number k and network coverage intensity t, which means the network coverage intensity C_n is no less than the threshold value t, we get the lower bound on the number of sensor nodes n satisfied to the application:

$$\begin{gathered} C_n \gg t, \\ 1 - \left(1 - \frac{q}{k}\right)^n \gg t, \\ \text{easily get, } 1 - t \gg \left(1 - \frac{q}{k}\right)^n, \\ \ln(1 - t) \ll n \ln\left(1 - \frac{q}{k}\right), \\ n \gg \frac{\ln(1 - t)}{\ln\left(1 - (q/k)\right)}. \end{gathered} \tag{3}$$

2.2. Enhanced Parallel Cat Swarm Optimization (EPCSO). In the field of swarm intelligence (AI), many optimization algorithms were being proposed in recent years. Chu et al. first present the cat swarm optimization algorithm [2] for solving optimization problems by model upon the natural behaviors of cats in 2006. In 2008, Tsai et al. propose a parallel cat swarm optimization (PCSO) algorithm based on the frame structure of parallelizing the CSO method, which has the ability to find the near best solution under more strict conditions. However, the operation in PCSO becomes more complex and requires more computational time to complete the whole process. Therefore, Tsai et al. propose a new algorithm called enhanced parallel cat swarm optimization in 2013 by adopting the orthogonal array of the Taguchi method into the tracing mode process of CSO. EPCSO algorithm has two modes called the seeking mode and the tracing mode for retaining the behaviors of cats to move the individuals in the solution space. The flowchart of EPCSO algorithm is given in Figure 1.

A brief review of EPCSO algorithm is given as follows.

Step 1. Create N cats, randomly classify the cats into the M-dimensional solution space within the limited ranges of the initial value, and randomly split them into G groups. Generate the velocities for each dimension of each cat and set the motion flags that define which mode the cat belongs to be according to the user predefined value of MR, where MR $\in$ [0, 1] (here, MR stands for the ratio of individuals moved by the seeking process and the tracing process, and MR also affects the ratio of artificial agents to work on the exploitation and exploration). The Seeking mode presents exploitative capacity. Tracing mode presents exploration capacity.

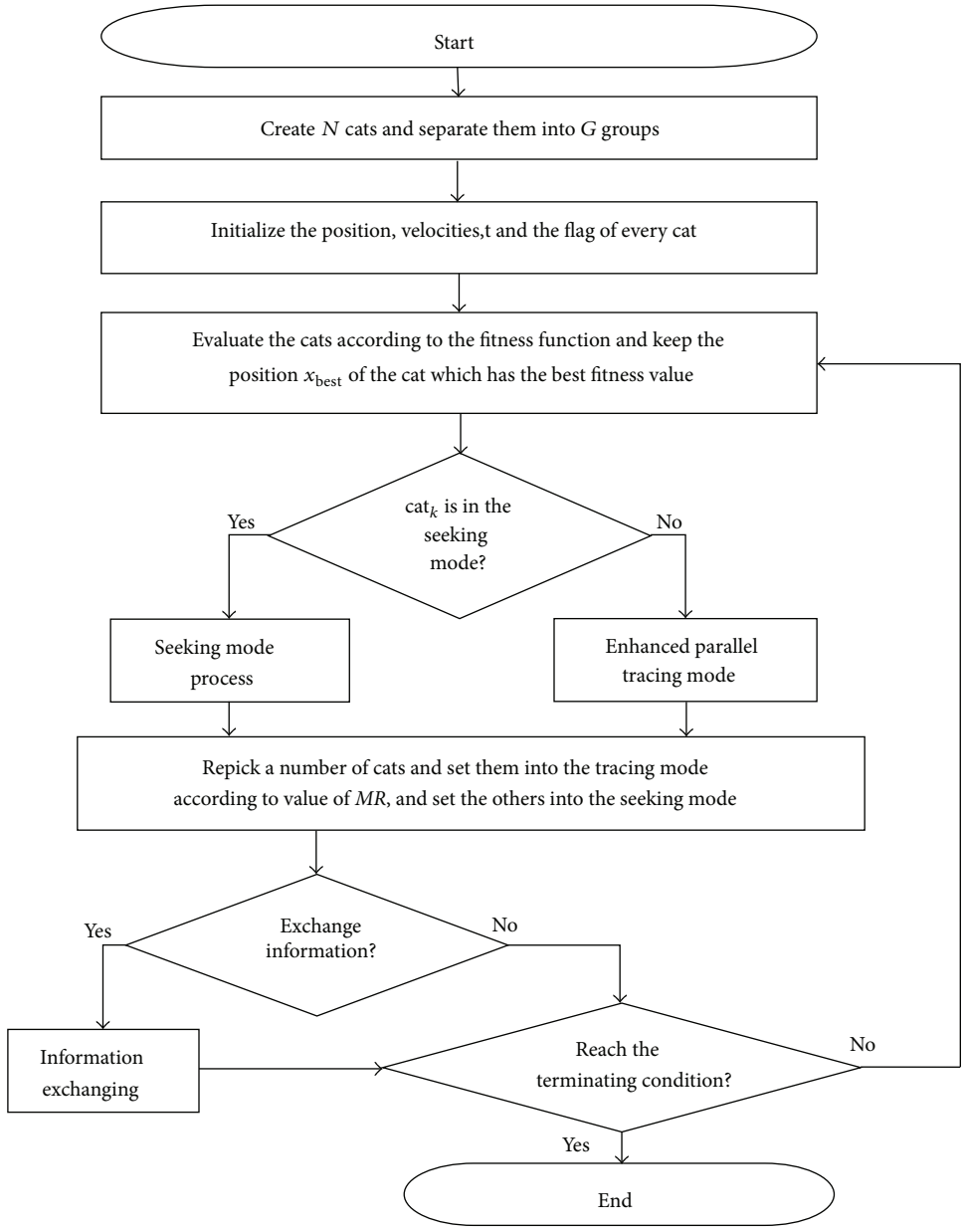

Figure 1: The flowchart of EPCSO algorithm.

Step 2. By taking the cats' coordinates into fitness function to evaluate the fitness values, respectively, record the best coordinate and the fitness value of the cat which has the better fitness value calculated so far.

Step 3. Move the cats by the seeking mode process or the tracing mode process. If the cat is assigned into seeking mode, it takes (4) and (5). Otherwise, it takes (6)–(9) according to the statues of the motion flag.

Step 4. Reset the motion flag for all cats. Repick and separate $[N \times (1 - \mathrm{MR})]$ cats into the tracing mode and the rest into the seeking mode.

Step 5. Check whether the number of iterations reaches a predefined iteration number or not. If it is satisfied, run the information exchanging process.

Step 6. Check whether the termination criteria are satisfied. If the answer is positive, output the coordinate, which represents the best solution found in the whole process and terminate the program. Otherwise, go back to Step 2 and repeat the process.

2.2.1. The Seeking Mode Process. Tsai et al. define 4 essential parameters in the seeking mode process, that is, the seeking memory pool (SMP), the seeking range of the selected

dimension (SRD), the counts of dimension to change (CDC), and the self-position considering (SPC). These parameters affect the searching ability, directly, because they are related to the quantity of the changes caused to the cats. The seeking mode process is briefly reviewed as follows.

Step 1. Generate SMP copies of the present position of cat. If SPC value is true, then store the present position as one of the candidates.

Step 2. For each copy cat, according to CDC, randomly plus or minus SRD percent values and replace the old ones:

$$x_j = (1 + \text{rand} \times \text{SRD}) \times x_j, \quad \text{where } \forall j. \tag{4}$$

Here rand is a random variable in the range [0-1].

Step 3. Calculate the fitness values (FS) of all candidate points, respectively.

Step 4. If all FS are not exactly equal, calculate the selecting probability P_i of each candidate by (5);

$$P_i = \begin{cases} 1, & \text{if } \text{FS}_{\max} = \text{FS}_{\min} \\ \dfrac{\left|\text{FS}_i - \text{FS}_b\right|}{\text{FS}_{\max} - \text{FS}_{\min}}, & \text{where } 0 < i < j, \text{ otherwise.} \end{cases} \tag{5}$$

If the goal of the optimization process is to find the minimum solution of the fitness function, let $\text{FS}_b = \text{FS}_{\max}$, otherwise let $\text{FS}_b = \text{FS}_{\min}$.

Step 5. Pick the coordinate from the candidate points to move to base on the probability.

2.2.2. The Parallel Tracing Mode Process. Once a cat runs into tracing mode, it migrates according to its' own velocities corresponding to every dimension. The parallel tracing mode process can be depicted as follows.

Step 1. Generate two sets of the velocities for every dimension $v_{k,d}(t)$ by (6) for the cat_k at the current iteration, marked as $CV_{1,d}(t)$ and $CV_{2,d}(t)$, respectively:

$$\begin{aligned} CV_{1,d}(t) &= V_{k,d}(t-1) + r \times c \\ &\quad \times \left[x_{g_{\text{best}},d}(t-1) - x_{k,d}(t-1)\right], \\ &\quad \text{where } d = 1, 2, \ldots, M, \\ CV_{2,d}(t) &= V_{k,d}(t-1) + r \times c \\ &\quad \times \left[x_{l_{\text{best}},d}(t-1) - x_{k,d}(t-1)\right], \\ &\quad \text{where } d = 1, 2, \ldots, M, \end{aligned} \tag{6}$$

where M denotes the dimension of the solution space, $x_{g_{\text{best}}}$ denotes the global near-the-best solution found so far, $x_{l_{\text{best}}}$ is the local near-the-best solution of the group. l represents the group that x_k belongs to, r is a random value in the range of [0-1], and c is a constant. Use the candidate velocities and the Taguchi orthogonal array to create a series of velocity sets, shown as follows:

$$v_{\text{set}_{s,d}}(t) = \begin{cases} cv_{1,d}(t), & \text{if the element in} \\ & \text{the orthogonal array is "0",} \\ cv_{2,d}(t), & \text{otherwise,} \end{cases} \tag{7}$$

where s is the index of the velocity set.

Step 2. Take one velocity set to update the original velocity $v_{k,d}(t)$, shown as follows:

$$v_{k,d}(t) = \begin{cases} v_{\max}, & \text{if } v_{k,d}(t-1) + v_{\text{set}_{s,d}}(t) \\ & \text{exceeds the} \\ & \text{maximum velocity,} \\ v_{k,d}(t-1) + v_{\text{set}_{s,d}}(t), & \text{otherwise.} \end{cases} \tag{8}$$

Then update the position $x_{k,d}(t)$ of the present cat_k as follows:

$$x_{k,d}(t) = x_{k,d}(t-1) + v_{k,d}(t). \tag{9}$$

Calculate its fitness value for later use. Accumulate the FS values contributed by the column factors and take the most adaptive factors to compose the latest velocity.

Step 3. Move the present cat_k with the latest velocity by (9) to update its position.

2.2.3. The Information Exchanging Process. This process aims at exchanging subpopulation information and forming the cooperation structure between different groups. There is a factor ECH to monitor the condition of each subpopulation. After ECH number iterations, these subpopulations exchange information once. Four steps of the exchanging process are showed as follows.

Step 1. Pick up a group of subpopulations sequentially and sort these cats according to their FS values.

Step 2. Randomly select a local best solution from an unrepeatable group.

Step 3. Replace the position of the worst of cat whose FS value is the worst with the local best solution cat position.

Step 4. Repeatedly perform Step 1 to 3 G times (there are only G groups) to let each group receive a local best solution from the others.

3. Network Model Deployment

In this paper, we consider two-dimensional static flat environment and assume each node equipped with the same fixed transmission range. The network region is a circle with radius 350. It would be much easier and cheaper than

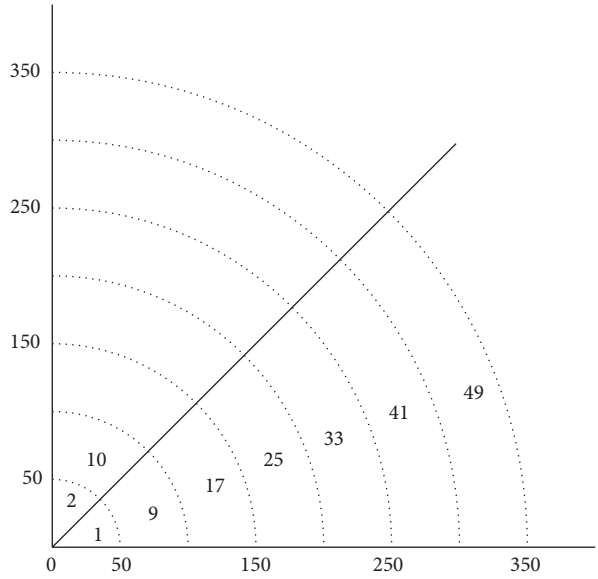

Figure 2: A quarter of network structure.

predefined position deployment. However, for the sake of diminishing the critical nodes workload and then prolonging the whole network lifetime, we propose a scheme for the sensor deployment. The closer to the sink node results in the more power consumption is a definite fact in deploying sensor nodes. The reason is that the nodes closer to the sink node are much frequently treated as the delay nodes. To overcome the power consumption problem, it is necessary to increase the number of nodes in the regions near to the sink node. Answering to the problem mentioned above, we propose a scheme to deploy the sensor nodes by calculating the required density of the sensor nodes. To deploy sensor nodes in a two-dimensional field, we divide the field into 7 concentric circles, 8 sectors, and 56 subboxes as shown in Figure 2.

There are 2,688 sensor nodes deployed in this field. The outer layer is defined as sectors on the 7th concentric circle; and the inner layer is defined as sectors numbered as 1 to 6 on the 1st concentric circle. For the outer layer concentric circles, each sector is assigned 48 sensor nodes scattered randomly to the field. Based on this condition, the same amount of sensor nodes is assigned to all sectors. This result in the coverage intensity is increased in the inner layer sectors. Therefore, the payload of WSN in the inner layer sectors can be shared with more optional paths. The power consumption is now with larger chance to be balanced automatically by deploying sensor nodes with higher density in the inner layer sectors. Furthermore, the possibility of network paralyzing caused by disabled internal nodes is reduced. In addition, the lifetime of the WSN is also extended. For a fixed amount of sensor nodes, the larger the measure of area is, the lower the coverage intensity is. To ensure the WSN is functional, the network coverage intensity must be above a certain threshold. Hence, in our deployment strategy, the minimum amount of sensor deployed in a sector is calculated based on the sector located on the outer layer. This deployment strategy is easy to operate. And the result is capable of achieving high coverage intensity. The proof of high coverage intensity is given as follows.

Proof. For the reason that the shape of each sector is not a circle, we cannot directly apply the minimum number of nodes theory to the sector area. Thus, we use the derivation method to verify the high coverage intensity. First, we calculate the coverage intensity value of an area covered by 48 sensor nodes. According to reverse deduction, the whole network minimum coverage intensity can be found.

(a) Only one subset exists in the WSN, $k = 1$. The network coverage intensity is t:

$$1 - \left(1 - \frac{a}{k}\right)^{s} \geq t, \tag{10}$$

where a is the area of circle with radius 350, which covers the whole region, and r denotes the size of sensing area of one node.

(b) It is known that the outer layer sector such as number 49, which is covered by 48 sensor nodes, satisfies the minimum full coverage condition. The same coverage intensity can be treated as the criterion for defining the number of sensor nodes used in other sectors to achieve the same coverage condition. The number of sensor nodes can be figured out by the equivalent proportional relationship. The area ratio of sector 49 to the whole field is $P_{\text{area}} = ((1 - ((\pi \times 300^2)/(\pi \times 350^2)))/8) = 13/392$ and $S = 48/(13/392) \approx 1{,}448$ is the number of sensor nodes for the circle with radius 350.

(c) In (10), given q, we get $q = r/a = \pi 50^2/\pi 350^2 = 1/49$.

(d) Finally, taking q, k, S into (10), we get

$$1 - \left(1 - \frac{1/49}{1}\right)^{1448} \geq t, \quad t \leq 0.9999999999995. \tag{11}$$

The area of a sector located in the outer layer such as sector 49 is larger than any sectors in the other layers. Hence, the coverage intensity of the most outer layer concentric is definitely smaller than the inner layer concentric sectors. In order to ensure a good data forwarding quality, the whole network coverage intensity is designed to be greater than 0.99. Based on the criteria mentioned above, the overview of the WSN based on our proposed deployment scheme is shown in Figure 3. □

4. Using EPCSO Method to Solve the Routing Problem in WSN

In the wireless sensor network, we need to build routing path for every sensor node. The routing path is different in the number of relay nodes for each concentric circle node. However, the EPCSO originally is not a method used in finding paths for the WSN. Its first application is designed for the aircraft schedule recovering. Thus, we have to partially modify EPCSO before employing it to solve the routing

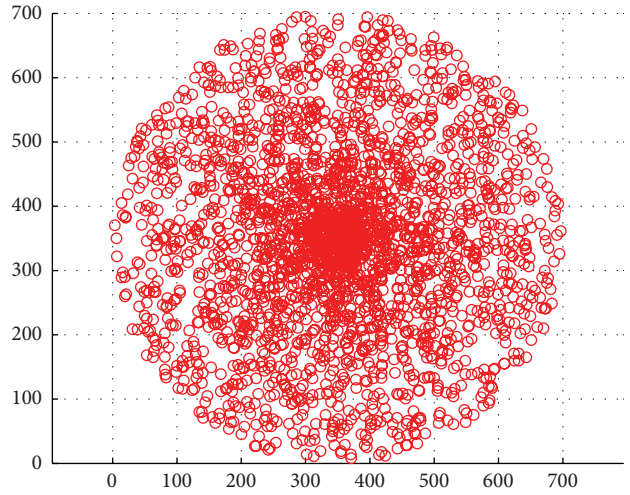

Figure 3: Network node deployment result.

problem in WSN. The modifications we made for EPCSO in this paper are listed as follows: (1) The representation of the artificial agent is modified from the coordinate to a set. (2) A newly defined cluster flag is added into the basic component of the artificial agent. (3) The newly designed fitness function is custom-made for the WSN routing problem.

Assume that the transmission range of a single sensor node is 50. Thus, the nodes located in the sectors in the 5th concentric circles need four relay nodes to transmit their package to the sink node. We use node *tar* as an example to explain how to use EPCSO to solve the routing problem. The modified EPCSO and the whole processes are explained as follows.

4.1. Initialization. In the initialization process, some parameters and constants are required to be defined before the whole process starts. In the original EPCSO, the artificial agent (the cat) is composed of the coordinate representing its position in the solution space. In our design, the artificial agent is composed by a set of sensor nodes. The sensor nodes in one set are from different layers and should be capable of forming at least one complete path from the most outer layer back to the sink node. For each cat, the set is defined by

$$\text{cat}_i = \{(x_{11}, x_{12}, x_{13}), (y_{21}, y_{22}, y_{23}), \ldots, (n_{l1}, n_{l2}, n_{l3})\}, \quad (12)$$

where i is the identity index of the cat and l is the identifier of the concentric circle (the layer). In this application, the population size is set to 16. There are three backup sensor nodes located in the same layer in every cat. As shown in Figure 4, sensor node *tar* in the given example is located on the 5th layer.

In this case, one of the initialization results of node *tar* can be described by

$$\text{cat}_1 = \{(a, b, c), (d, e, f), (i, g, h), (m, j, k)\}. \quad (13)$$

As given in (13), cat_1 consists of nodes in 4 layers, where sensor nodes a and b belong to the first layer and sensor

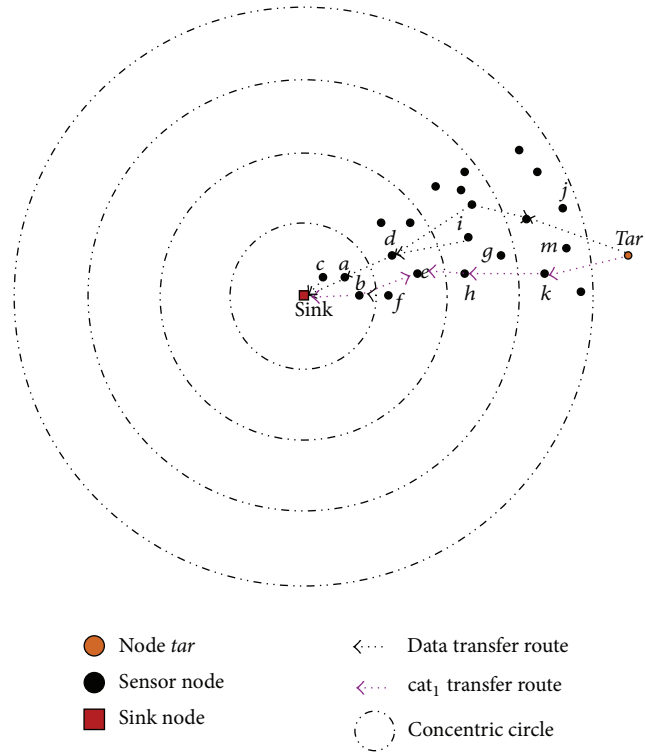

Figure 4: One of the initialized cats for node *tar*.

nodes d, e, and f are located in the second layer and so on. There is only one integrated routing path in cat_1; routing path $\{k, h, e, b\}$ is composed of four sensor nodes. Each node is collected from different layers, and the order of path is exactly the relay node order for node *tar* to transmit package to the sink node. In addition, all sensor nodes employed in one cat should be gathered from the same quadrant. This criterion is capable of avoiding the incomplete path caused by a sudden jump, which exceeds the sensing range of the sensor node. Besides the set of sensor nodes, a cat should also carry the corresponding velocities to all components. The velocity of a cat is described by

$$V_{\text{cat}} = \{(v_{11}, v_{12}, v_{13}), (v_{21}, v_{22}, v_{23}), \ldots, (v_{l1}, v_{l2}, v_{l3})\}. \quad (14)$$

The velocity is corresponding to the composition of cat. For example, v_{11} plays the role of the velocity for the sensor node a. The velocities are constrained within a predefined maximum velocity for every dimension. The maximum velocity set is defined by

$$V_{\max} = \{v_1, v_2, v_3, \ldots, v_l\}, \quad (15)$$

where l stands for the numerical value of concentric circles and $v_1 < v_2 < v_3 < \cdots < v_l$. The velocity increases gradually because the area of a sector increases in the outer layers. The last step in the initialization process is to set the motion flag and the cluster flag for every cat. The cluster flag is defined to indicate the cluster number which the cat belongs to. Based on our design, the 2D space is divided into 8 sectors in every layer. Hence, the cluster flag is an integer in the range of [1, 8].

In every cluster, we can find at least one cat, which presents the best fitness value in its own cluster. This cat will be marked as the local best solution in the cluster and is denoted by C_{best_I}, where I stands for the cluster label.

4.2. Custom-Made Fitness Function. The fitness function (also called the object function or the evaluation function) plays the principal role in the whole process. A well-designed fitness function should be capable of representing the input solution's behavior in the solution space.

To utilize the modified EPCSO in finding the balanced path for the WSN, a fitness function is designed specifically for this goal. Our proposed fitness function can be described by

$$F_x = \alpha_1 \frac{x_1}{3^l} + \alpha_2 \frac{x_2}{x_1} + \alpha_3 \frac{1 - x_3}{3^{l-1}} + \alpha_4 \frac{x_4}{3 \cdot l}, \tag{16}$$

where x_1 denotes the sum of all collected paths which go through the current node in the cat, x_2 stands for the average power consumption of all connected paths collected in the cat, x_3 is the relay counter for calculating the number of forwarding packages for other nodes, x_4 is used for counting the number of usage node existing in the cat, and α_1, α_2, α_3, and α_4 are the weights and $\sum_{i=1}^{4} \alpha_i = 1$. The parameters in the fitness function are described in detail as follows.

(a) *Path Number* (x_1). This parameter is the accumulation of collected paths which go through the current node. However, the sensor nodes are not allowed to transmit packages to their neighborhood nodes located in the same layer. The transmission path between nodes in the same layer should be eliminated.

(b) *Total Power Consumption* (x_2). This parameter accumulates all power consumption caused by paths carried by this cat.

(c) *Relay Number* (x_3). This parameter indicates the count of legal transmission between neighborhood nodes. As mentioned above, the transmission between nodes located in the same layer is forbidden.

(d) *Usage Number* (x_4). This parameter counts the number of nodes in the cat involved in building the integrated routing path from the current node to the sink node. These nodes are called the useful nodes.

The fitness function can be decomposed into 4 parts, that is, the path ratio, the average power consumption, the relay ratio, and the usage rate. The detailed description is listed as follows.

(a) *Path Ratio*. The path ratio is defined by (17); it stands for the ratio of the found path numbers to the ideal path in the cat:

$$\text{Path}_{\text{ratio}} = \frac{x_1}{3^l}, \tag{17}$$

where $\text{Path}_{\text{ratio}}$ represents the path ratio, l is the position circles minus one, and the ideal path number is known as 3^l.

(b) *Average Power*. The average power is defined by (18). $\text{Ave}_{\text{power}}$ calculates the power consumption on average over all paths carried by the cat:

$$\text{Ave}_{\text{power}} = \frac{x_2}{x_1}. \tag{18}$$

(c) *Relay Ratio*. The relay ratio is defined by (19). It is known that, in the ideal case, the total relay number is 3^{l-1}. Hence, the relay ratio is equal to 1 in the ideal case:

$$R = \frac{x_3}{3^{l-1}}. \tag{19}$$

(d) *Usage Rate*. The usage rate is defined by (20). It stands for the ratio of the usable nodes to all nodes collected by the cat. In the ideal case, all nodes carried by a cat should be useable; that is, the usage rate is equal to 1:

$$U = \frac{x_4}{3 \cdot l}. \tag{20}$$

The fitness function is employed in the EPCSO process to evaluate the fitness of the cats and is the gauge to find the global near-the-best solution (denoted by G_{best}) and the local near-the-best solution (denoted by G_{best_I}) for the Ith cluster.

4.3. An Example Is Given as Follows. Assume the target node *tar* is located in the 5th layer as shown in Figure 4; the parameter l is 4. For cat_1, we have $x_1 = 1$, and $\text{Ave}_{\text{power}}$ is the average power consumption of path $\{k, h, e, b\}$. $R = 7$ because the relay ratio is the set of $\{(j, g), (k, h), (e, h), (b, e), (b, f), (d, i), (a, d)\}$, and $U = 4$ because nodes $\{k, h, e, b\}$ are usable.

4.4. Seeking Mode Process. Suppose cat_1 is assigned to move by the seeking mode process; j copies of cat_1 are made in the beginning. If the SPC is set to "true" by the user, the copies should remain one set identical to cat_1. Based on the value or CDC and SRD, the rest of the copies are slightly modified. An example is given as follows. Suppose we have cat_1 in the process and cat_1 is given in

$$\text{cat}_1 = \{(a, b, c), (d, e, f), (i, g, h), (m, j, k)\}. \tag{21}$$

Assume CDC = 2; nodes m and i are chosen to be the mutative nodes as shown in Figure 5. The node index is modified by SRD percents. After the process, it is possible that node m is changed to node n; and node i is changed to node p as shown in Figure 6. Calculate and sort the fitness values of all copies by (16) and keep the best cat in the memory.

4.5. Tracing Mode Process. In the tracing mode process, the cat's velocity should be updated, and the cat will be moved based on its velocity produced by the Taguchi method. The velocity is produced based on G_{best} and G_{best_I}. We take the same example listed in Figure 5. Let cat_1 be initialized for node *tar*; we get $CV_{1,d}(t)$ and $CV_{2,d}(t)$ two

Table 1: The $L_{13}(2^{12})$ orthogonal array.

Experiment number	Considered factors											
	A	B	C	D	E	F	G	H	I	J	K	L
1	−1	−1	1	−1	1	1	−1	−1	1	1	−1	1
2	1	−1	−1	−1	−1	1	1	−1	−1	1	1	1
3	−1	1	−1	−1	1	−1	1	−1	1	−1	1	1
4	1	1	1	−1	−1	−1	−1	−1	−1	−1	−1	1
5	−1	−1	1	1	−1	−1	1	−1	1	1	−1	−1
6	1	−1	−1	1	1	−1	−1	−1	−1	1	1	−1
7	−1	1	−1	1	−1	1	−1	−1	1	−1	1	−1
8	1	1	1	1	1	1	1	−1	−1	−1	−1	−1
9	−1	−1	1	−1	1	1	−1	1	−1	−1	1	−1
10	1	−1	−1	−1	−1	1	1	1	1	−1	−1	−1
11	−1	1	−1	−1	1	−1	1	1	−1	1	−1	−1
12	1	1	1	−1	−1	−1	−1	1	1	1	1	−1
13	−1	−1	1	1	−1	−1	1	1	−1	−1	1	1
14	1	−1	−1	1	1	−1	−1	1	1	−1	−1	1
15	−1	1	−1	1	−1	1	−1	1	−1	1	−1	1
16	1	1	1	1	1	1	1	1	1	1	1	1

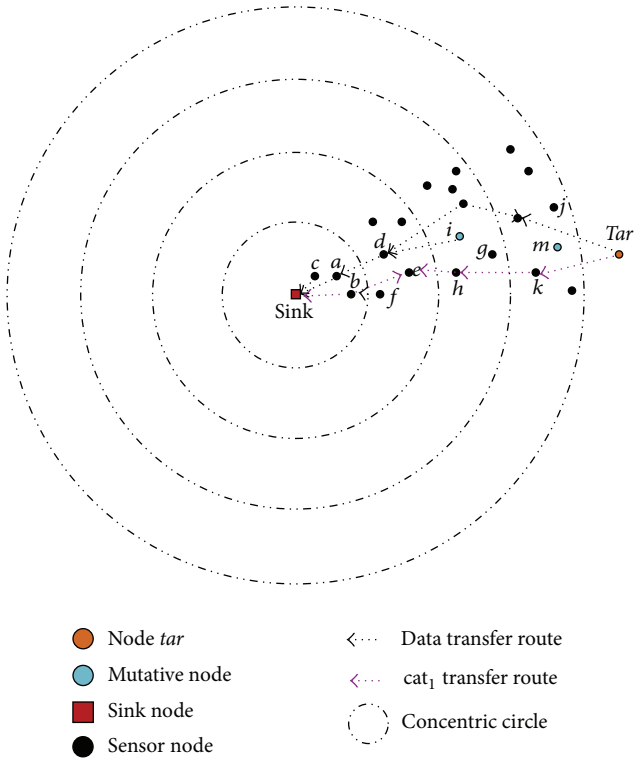

Figure 5: An example of cat_1 in the tracing mode process.

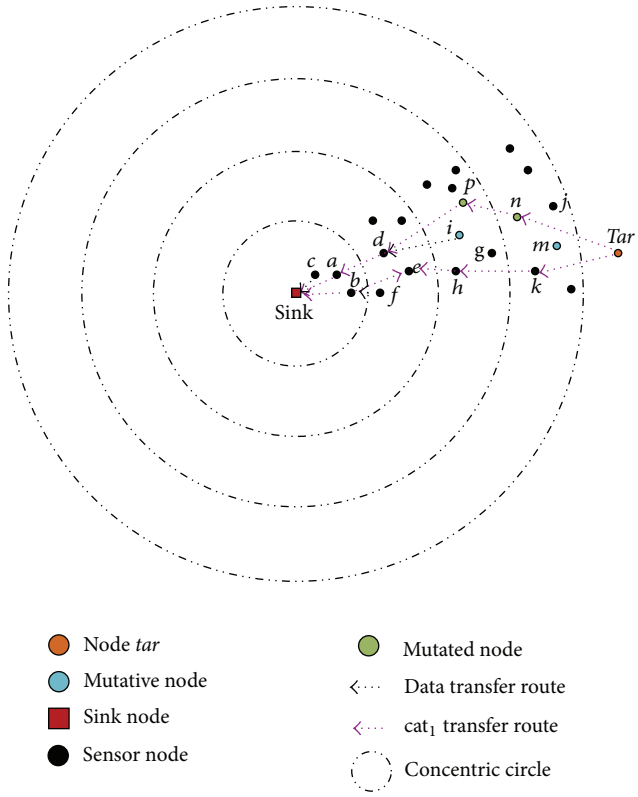

Figure 6: The result of cat_1 after the tracing mode process.

twelve-dimensional velocities sets. To find the optimal combination of the velocity, EPCSO uses Taguchi method to solve this problem. Twelve dimensional velocities mean twelve column factors. The $L_{13}(2^{12})$ orthogonal array is given in Table 1. The values "−1" and "1" in the elements of the orthogonal array indicate the value from $CV_{1,d}(t)$ or $CV_{2,d}(t)$ should be used in the corresponding dimension.

5. Experiment and Experimental Result

The experimental result is produced by our proposed routing method based on the minimum-number-node theory and the modified EPCSO. As mentioned above, 2,688 nodes are deployed in a 2D field within a 350-radius circle. The

Table 2: The results of four algorithms.

Nodes	Methods	All	Average	Max	Min
All	LD_{ACO} (Ho et al. 2012) [10]	3.5417×10^6	1.317×10^3	3.10×10^4	6.7×10^1
	AODV (Perkins and Royer, 1999) [6]	3.5772×10^6	1.330×10^3	8.82×10^4	6.7×10^1
	LD_{CSO} (Kong et al., 2014) [7]	3.5881×10^6	1.334×10^3	5.15×10^4	6.7×10^1
	Our proposed method	2.2984×10^6	0.855×10^3	1.41×10^4	6.7×10^1
One	LD_{ACO} (Ho et al. 2012) [10]	6.7614×10^5	1.760×10^3	3.10×10^4	6.7×10^1
	AODV (Perkins and Royer, 1999) [6]	7.0002×10^5	1.822×10^3	8.82×10^4	6.7×10^1
	LD_{CSO} (Kong et al., 2014) [7]	6.9459×10^5	1.808×10^3	5.15×10^4	6.7×10^1
	Our proposed method	2.9920×10^5	7.99×10^2	1.41×10^4	6.7×10^1
Two	LD_{ACO} (Ho et al. 2012) [10]	5.7792×10^5	1.505×10^3	1.95×10^4	6.7×10^1
	AODV (Perkins and Royer, 1999) [6]	5.8802×10^5	1.531×10^3	3.71×10^4	6.7×10^1
	LD_{CSO} (Kong et al., 2014) [7]	5.9846×10^5	1.558×10^3	4.12×10^4	3.5×10^1
	Our proposed method	2.3038×10^5	6.01×10^2	7.14×10^3	9.0×10^1
Six	LD_{ACO} (Ho et al. 2012) [10]	1.8860×10^5	4.91×10^2	2.81×10^3	5.7×10^1
	AODV (Perkins and Royer, 1999) [6]	1.7568×10^5	4.57×10^2	9.44×10^3	1.2×10^1
	LD_{CSO} (Kong et al., 2014) [7]	1.7617×10^5	4.58×10^2	8.02×10^3	1.2×10^1
	Our proposed method	4.7954×10^5	1.248×10^3	3.29×10^3	2.5×10^2
Seven	LD_{ACO} (Ho et al. 2012) [10]	7.4627×10^4	1.94×10^2	7.09×10^2	5.8×10^1
	AODV (Perkins and Royer, 1999) [6]	8.4859×10^4	2.20×10^2	3.23×10^3	1.5×10^1
	LD_{CSO} (Kong et al., 2014) [7]	8.8532×10^4	2.26×10^2	1.91×10^3	3.1×10^1
	Our proposed method	3.0722×10^5	8.00×10^2	3.87×10^3	2.3×10^2

circle is divided into 7 concentric circles and each of the concentric circles contains the same number of sensor nodes, that is, 384 nodes. The experimental result of the total power consumption produced by our proposed method is compared with LD_{ACO} [10], AODV [6], and LD_{CSO} [7] in Table 2. In our simulation, we only count the routing power consumption. The routing path is from one node that senses an event to the sink. In the first column, "*all*" stands for the power consumption caused by all sensor nodes deployed in the environment, "*one*" means the power consumption caused by all sensor nodes located in the most inner layer, "*two*" is the power consumption caused by the sensor nodes located in the 2nd layer, and "*seven*" is the power consumption caused by the sensor nodes located in the most outer layer. The experimental result indicates that the whole power consumption is 3.5417×10^6 for LD_{ACO} algorithm, and the average power consumption for one node is 1.317×10^3, the maximum consumption is 1.310×10^4, and the minimum is 6.7×10^1. The above power consumption is for all the nodes. The rest of rows results are for each of the concentric circles sensor nodes, respectively. The proposed method gets better performance due to low total power consumption and balance of consumption for the in-layer nodes and outer layer nodes. This could partially solve the problem for prolonging the lifetime of sensor network.

6. Conclusion

In this paper, we propose a strategy for deploying the sensor nodes by considering the coverage of WSN based on the minimum-number-node theory. Furthermore, we utilize EPCSO to design a routing algorithm for providing the balanced routing paths. Three modifications are made for the EPCSO algorithm to make it suitable for finding routing paths for WSN. Our design balances the power consumption between the forwarding distance and the energy saving for all nodes in the whole WSN. Our deployment strategy is easy to use in different WSN applications. Furthermore, the problem of which the inner layer nodes usually are out of battery is solved because our deployment method with EPCSO finding transmission paths keeps the sensor nodes in the inner layer alive and the nodes in the most outer layer are firstly out of battery.

The experimental result indicates that our proposed method could partially balance the power consumption between the inner layer and the outer layer. The total power consumption of our proposed method is the lowest among other algorithms. The simulation results indicate that our proposed method reduces more than 35% power consumption on average.

Conflict of Interests

The authors declare that there is no conflict of interests regarding the publication of this paper.

References

[1] M. Prauzek, P. Musilek, A. G. Watts, and M. Michalikova, "Powering environmental monitoring systems in arctic regions: a simulation study," *Elektronika ir Elektrotechnika*, vol. 20, no. 7, pp. 34–37, 2014.

[2] S. C. Chu, P. W. Tsai, and J. S. Pan, "Cat swarm optimization," in *PRICAI 2006: Trends in Artificial Intelligence: roceedings of*

the 9th Pacific Rim International Conference on Artificial Intelligence, Guilin, China, vol. 4099 of *Lecture Notes in Computer Science*, pp. 854–858, Springer, Berlin, Germany, 2006.

[3] P.-W. Tsai, J.-S. Pan, S.-M. Chen, and B.-Y. Liao, "Enhanced parallel cat swarm optimization based on the Taguchi method," *Expert Systems with Applications*, vol. 39, no. 7, pp. 6309–6319, 2012.

[4] P. M. Pradhan and G. Panda, "Solving multiobjective problems using cat swarm optimization," *Expert Systems with Applications*, vol. 39, no. 3, pp. 2956–2964, 2012.

[5] P.-W. Tsai, J.-S. Pan, S.-M. Chen, B.-Y. Liao, and S.-P. Hao, "Parallel cat swarm optimization," in *Proceedings of the 7th International Conference on Machine Learning and Cybernetics (ICMLC '08)*, pp. 3328–3333, Kunming, China, July 2008.

[6] C. E. Perkins and E. M. Royer, "Ad-hoc on-demand distance vector routing," in *Proceedings of the 2nd IEEE Workshop on Mobile Computing Systems and Applications (WMCSA '99)*, pp. 90–100, February 1999.

[7] L. P. Kong, C. M. Chen, H. C. Shih, C. W. Lin, B. Z. He, and J. S. Pan, "An energy aware routing protocol using cat swarm optimization for wireless sensor networks," in *Advanced Technologies, Embedded and Multimedia for Human centric Computing*, vol. 260, pp. 314–318, Springer, 2014.

[8] J.-W. Lin and Y.-T. Chen, "Improving the coverage of randomized scheduling in wireless sensor networks," *IEEE Transactions on Wireless Communications*, vol. 7, no. 12, pp. 4807–4812, 2008.

[9] C. Liu, K. Wu, Y. Xiao, and B. Sun, "Random coverage with guaranteed connectivity: joint scheduling for wireless sensor networks," *IEEE Transactions on Parallel and Distributed Systems*, vol. 17, no. 6, pp. 562–575, 2006.

[10] J.-H. Ho, H.-C. Shih, B.-Y. Liao, and S.-C. Chu, "A ladder diffusion algorithm using ant colony optimization for wireless sensor networks," *Information Sciences*, vol. 192, pp. 204–212, 2012.