Aiming at the perception hole caused by the necessary movement or failure of nodes in the wireless sensor actuator network, this paper proposed a kind of coverage restoring scheme based on hybrid particle swarm optimization algorithm. The scheme first introduced network coverage based on grids, transformed the coverage restoring problem into unconstrained optimization problem taking the network coverage as the optimization target, and then solved the optimization problem in the use of the hybrid particle swarm optimization algorithm with the idea of simulated annealing. Simulation results show that the probabilistic jumping property of simulated annealing algorithm could make up for the defect that particle swarm optimization algorithm is easy to fall into premature convergence, and the hybrid algorithm can effectively solve the coverage restoring problem.

1. Introduction

Wireless Sensor and Actuator Network, WSAN, is an ad hoc and multihop network system, which is composed of a large quantity of microsensor nodes, laid in unattended monitoring areas, and formed by wireless communication method. The purpose of the system is to provide detailed and accurate information to remote observer, by cooperatively sensing, connecting, and dealing with perceived object information in monitoring areas and then processing them [1]. Since the sensor nodes are with some real constraints, nodes energy is limited; moreover it is unrealistic to supplement energy by replacing battery; nodes communication capability is limited, and so forth, so network coverage research can be taken as to optimize allocation of various limited resources within sensor network through node deployment and node position adjustment, hence to effectively improve various services quality: environmental perception, information acquisition, data transmission and network survival ability, and so forth.

Moreover, after the network running for some time, due to some nodes energy depletion or physical damage, and so forth, network awareness and communication capabilities are reduced, which results in coverage hole, thereby affecting network coverage. Coverage hole will cause perceptual information incompleteness, reduce information validity, and make network communication not smooth, thus affecting whole network performance. To fulfill related task and realize its value, firstly, wireless sensor actuator network must cover monitoring areas well. Network coverage is an important indicator to judge wireless sensor actuator network performance and service quality.

Currently, coverage restoring unconstrained optimization has two categories, wherein one is to add nodes in network hole and another is to move existing nodes. Wang and Wu [2] propose a distributed hole detection and restoring method under trap coverage, wherein hole detection method enables node to confirm hole position autonomously; then by adding nodes in hole, network coverage increases. Lu et al. [3] use particle swarm optimization algorithm to find sink node position in randomly arranged network and then increase network coverage and optimize topology of network. Aiming at coverage restoring problem in dense distributed wireless sensor network, Yang et al. [4] propose coverage restoring algorithm SOI, which calculates nodes moving direction and best position to move on hole edge, and then expand nodes coverage by nodes moving, hence fulfilling network coverage restoring. Lee et al. [5] propose CRAFT with certain fault tolerance to solve network coverage problem,
which establishes two-way connection zoning topology, builds the largest backbone around damaged area center, and then deploys the relay nodes (RNs) to connect each partition to the backbone outer polygon. Simulation states that this algorithm can get topology with high coverage by placing less relay nodes. With distributed self-deployment strategy, Senturk et al. [6] recover continuity of zone wireless sensor network and proposes two methods to locate relay nodes, wherein the first one is based on movement of relay nodes virtual force and the second one is based on game theory in zone guiding nodes. Simulation states that, in most situations, the method based on game theory performs better than the one on virtual force. Aiming at the coverage decrease problem due to the failure of many nodes in network, Zhao and Wang [7] propose a flow-based multiobjective nonlinear mathematical programming model, which decreases coverage by moving nodes in network, takes the total travel distance of nodes and single travel distance as optimization target, and then uses flow equilibrium conditions to restore network coverage.

As current improvement on integrated circuit technology leads to lower energy consumption on sensor and processor, relatively, the communication energy consumption between nodes becomes too low to consider. So when calculating the energy consumption, the communication energy consumption can be ignored. Move energy consumption is defined as the energy consumed when attack node moves towards best attack point, so move energy consumption is proportional to the distance that attack nodes moves in; hence distance that attack nodes moves in can be used to calculate move energy consumption directly.

2. Models for Problem Solving

2.1. Node Sensing Model. Node Sensing Model [8, 9] is a prior problem in network coverage control technology. Three main perception models have been used in recent research work, which include Circumference Sensing Model, Probability Sensing Model, and Direction Sensing Model. Circumference Sensing Model defines nodes sensing range as a round enclosed area which takes nodes location as center and sensing radius $R_s$ as radius. The sensing radius $R_s$ is decided by physical characteristics of the sensor. Any point in this round area can be sensed by node.

Circumference Sensing Model assumes that uncertainty does not exist when the sensor senses target area. But, in practical application, due to the disruptive factors such as ambient noise, obstacle, and the feature that wireless signal intensity decays along with the communication distance increase, circumference Sensing Model cannot reflect nodes sensing feature well. Probability Sensing Model considers the uncertainty of node sensing in practical application, wherein the probability that an intruder $d$ sensed by node $n$ is expressed as

$$p_s(n,d) = \begin{cases} 1, & R_s - r \geq \|(x_n, y_n) - (x_d, y_d)\| \leq R_s + r \\ e^{-\lambda \alpha d}, & R_s - r \leq \|(x_n, y_n) - (x_d, y_d)\| \leq R_s + r \\ 0, & R_s + r \leq \|(x_n, y_n) - (x_d, y_d)\|. \end{cases} \quad (1)$$

In this formula, $R_s$ is for the max sensed radius without disruptive factors; $r$ is for measuring nodes uncertain monitoring ability. Arguments $\alpha = \|(x_n, y_n) - (x_d, y_d)\| - (R_s - r)$, wherein $\lambda$ and $\beta$ are for intruder sensed probability value by node when intruder is in uncertain monitoring area. Physical characteristic of Probability Sensing Model is as shown in Figure 1.

Sensing direction of the two Sensing Models is omnidirectional. But, in practical application, some sensors have certain direction, wherein only when intruder is in certain direction of nodes can they be sensed. The Node Sensing Model setup under this feature is called Directed Sending Model.

2.2. Related Definitions

Definition 1 (effective coverage area). Supposing an intruder can be sensed by node at any point in designated area, then the designated area is called coverage area $\psi(i)$ of node $i$. Effective coverage area $i$ is defined as intersection of coverage area $\psi(i)$ and monitor area $\Omega$:

$$\xi(i) = \psi(i) \cap \Omega. \quad (2)$$

Definition 2 (coverage ratio). Coverage ratio of node $i$ is defined as the area ratio of effective node coverage area $\xi(i)$ to monitor area $\Omega$. Network coverage ratio is defined as the area ratio of the union of all node effective coverage areas in monitor area. Network coverage ratio is an important indicator of the quality of network coverage:

$$\eta = \frac{S(\bigcup \xi(i))}{S(\Omega)}. \quad (3)$$

Definition 3 (connectivity). If two nodes can communicate through single-hop or multi-hop network, then they are called connected. Supposing that the total number of nodes in monitor area is $n$, obviously, the nodes number $\xi(i)$ connected

![Figure 1: Probability Sensing Model.](image-url)
with node \( i \) will satisfy the formula \( 0 \leq \xi(i) \leq n - 1 \). If all nodes can communicate between each other, then the total number of communication paths is \( n(n - 1)/2 \). Connectivity of network is defined as the total number ratio of current communication path to max communication path:

\[
\tau = \frac{1}{2} \cdot \sum \xi(i) = \frac{\sum \xi(i)}{n(n-1)}.
\]

When doing research on network coverage problem, the paper supposes that network and nodes have the following characteristics:

1. Nodes in network are isomorphic and movable and have unique short addresses to identify themselves.

2. Sensing model of nodes is Probability Sensing Model, which has the same sensing radius \( R_s \) and communication radius \( R_c \), and meets \( R_c \geq 2 \cdot R_s \). When it meets this condition, network coverage problem is of equal value to connection problem.

Monitor area is a square area, with side length \( L \), and \( N \) nodes are randomly and evenly deployed in monitor area.

3. Each node gets its own location information from self-positioning of network and then broadcasts to the entire network.

2.2. Network Coverage Based on Grids. Ideally, when any point in monitor area is in node sensing range, then area is totally covered, and network coverage ratio is 1. In practical application, as nodes are deployed randomly, it is possible that effective coverage areas will multioverlap as shown in Figure 2.

Approximate calculation method on network coverage is as follows: dividing monitor area into equal grids, if grids are small enough, then node coverage to grid approximates to that of to grid center point. In this case, network coverage ratio approximates to nodes coverage to all grid center points.

If in monitor area, grid number is \( A_{gr} \), and node number is \( A_n \). According to Probability Sensing Model, the sensing probability of center point of grids \( G_i \) sensed by nodes \( N_j \) is

\[
p_{g,n}(G_i,N_j) = \begin{cases} 
1, & R_s - r \geq d(G_i,N_j) \\
e^{-\lambda \alpha^\gamma}, & R_s - r < d(G_i,N_j) < R_s + r \\
0, & R_s + r \leq d(G_i,N_j),
\end{cases}
\]

wherein \( d(G_i,N_j) \) is for distance between center point of grids \( G_i \) and nodes \( N_j \), \( R_s \) is for the max sensed radius of nodes without disruptive factors, and parameters \( r, \lambda, \alpha, \beta \) have the same meaning as in formula (1).

As it is an independent event that whether center point of grids \( G_i \) can be sensed by nodes \( N_j \), or not, according to probability theory, the probability that grids \( G_i \) are sensed by at least one node is

\[
p_g(G_i) = 1 - \prod_{j=1}^{A_n} \left( 1 - p_{g,n}(G_i,N_j) \right).
\]

This paper, the probability that all center points of grids \( G_i \) in monitor area are sensed by at least one node is approximated to network coverage ratio \( \eta \):

\[
\eta = \frac{\sum_{i=1}^{A_g} p_g(G_i)}{A_g}.
\]

3. Particle Swarm Optimization Algorithm

The network coverage problem can be abstracted as the optimization goal of the network coverage, and the non-constrained optimization problem of the decision variables is based on the coordinate of the node. In this paper, the particle swarm optimization algorithm is used to solve the optimization problem. Because the coordinate value of the decision variable node is continuous, it is different from the task assignment problem, and the network coverage problem needs the continuous particle swarm optimization algorithm. Since the particle swarm optimization algorithm has been proposed, many researchers have put forward many improved algorithms based on different practical applications.

3.1. Basic Particle Swarm Optimization Algorithm. Particle swarm optimization algorithm is designed from research on bird flock preying behavior. Assuming that there is only one piece of food in a searching area and that bird flock is randomly distributed in this area without knowing where the food is and how far away from its location, its task is to find that food. Each individual in the bird flock updates its current position according to history information of individual as well as group. By updating location constantly, the bird flock confirms the exact location of food, thus completing preying task. The researcher is inspired by bird flock preying model; hence particle swarm optimization algorithm is proposed to
solve optimization problem. There are solutions in research area, wherein particle swarm is randomly located in, and each particle is a potential solution to optimization problem. These particles are evaluated by fitness value which is decided by optimization target function, and each particle decides its own flying speed information according to history best fitness value of its own as well as group and then moves at this speed in entire solution space; that is, the individual exchanges value of its own as well as group and then moves at this speed in entire solution space; that is, the individual exchanges value of its own as well as group and then moves at this speed in entire solution space; that is, the individual exchanges value of its own as well as group and then moves at this speed in entire solution space; that is, the individual exchanges value of its own as well as group and then moves at this speed in entire solution space; that is, the individual exchanges value of its own as well as group and then moves at this speed in entire solution space; that is, the individual exchanges value of its own as well as group and then moves at this speed in entire solution space; that is, the individual exchanges value of its own as well as group and then moves at this speed in entire solution space.

Mathematical description on basic PSO algorithm is as follows.

Assuming that population size of particle swarm is $n$, decision space is $m$, wherein the location of particle $I$ at time $t$ denotes $X^{(t)}_i = (x^{(t)}_{i1}, x^{(t)}_{i2}, \ldots, x^{(t)}_{im})$, $i = 1, 2, \ldots, n$, and speed of particle $I$ is defined as particle moving distance in each iteration, which is denoted as $V^{(t)}_i = (v^{(t)}_{i1}, v^{(t)}_{i2}, \ldots, v^{(t)}_{im})$, $i = 1, 2, \ldots, n$; hence the moving speed and location of particle $I$ at time $t + 1$ in $d$ ($d = 1, 2, \ldots, m$) space change according to the formulas below [10]:

$$
v^{(t+1)}_{id} = w \cdot v^{(t)}_{id} + c_1 \cdot r_1 \cdot (p^{(t)}_{id} - x^{(t)}_{id}) + c_2 \cdot r_2 \cdot (g^{(t)}_{id} - x^{(t)}_{id}), \quad (8)
$$

$$
v^{(t+1)}_{id} = \begin{cases} v^{\max}_{id}, & v^{(t+1)}_{id} > v^{\max}_{id} \\ -v^{\max}_{id}, & v^{(t+1)}_{id} < -v^{\max}_{id} \\ v^{(t+1)}_{id}, & \text{otherwise} \end{cases}, \quad (9)
$$

$$
x^{(t+1)}_{id} = x^{(t)}_{id} + v^{(t+1)}_{id}, \quad (10)
$$

wherein, in formula (8), $w$ is for inertia weight, which is mainly for producing disturbance to prevent premature convergence on algorithm; $c_1$ and $c_2$ are for acceleration constants, which adjust maximum step size of particle moving to the best individual particle and the best global particle; and $r_1$ and $r_2$ are two random numbers in range $[0, 1]$, $p^{(t)}_{id}$ is for $d$th-dimensional component of individual extremum $p_{best}$, $g^{(t)}_{id}$ is for $d$th-dimensional component of global extremum $g_{best}$. In formula (9), $v^{\max}_{id}$ is for particle max flight speed, which is a constant, and is used to limit particle max flight speed to improve searching result. As shown in formulas (8)–(10), particle moving velocity increment is closely related to its own history flying experience and group flying experience and limited by max flight speed [11].

3.2. Particle Swarm Optimization Algorithm with Compression Factor. Learning factors $c_1$ and $c_2$ and the particles having a self-summary to the group of outstanding individual learning ability, respectively, this reflects the exchange of information between the particle swarm. If $c_1$ is larger, the particle will make more wandering in the local area, and if $c_2$ is larger, the particle will prematurely be converged as a local minimum value.

In order to control the particle speed effectively, making the algorithm balanced between global and local optimization, Clerc and Kennedy [12] proposed a constriction factor PSO algorithm, and the speed of the particle update formula will be changed:

$$
v^{(t+1)}_{id} = \varphi \cdot \begin{cases} v^{(t)}_{id} + c_1 \cdot r_1 \left( \frac{p^{(t)}_{id} - x^{(t)}_{id}}{v^{(t)}_{max}} \right) + c_2 \cdot r_2 \left( \frac{g^{(t)}_{id} - x^{(t)}_{id}}{v^{(t)}_{max}} \right), \quad (11)
\end{cases}
$$

$$
\varphi = \frac{2}{\left[ 2 - C \sqrt{C^2 + 4C} \right]}, \quad C = c_1 + c_2.
$$

In order to ensure the solution of the algorithm, $c_1 + c_2$ value must be greater than 4. Typical parameters are as follows:

1. $c_1 = c_2 = 2.05, C = 4.1$, and shrinkage factor $\varphi$ is 0.729.
2. (2) Particle population size $n_{pop} = 30, c_1 = 2.8, c_2 = 1.3, C$ is 4.1 at this time, and the shrinkage factor $\varphi$ is 0.729.

3.3. Particle Swarm Optimization Algorithm with Improved Weight. Inertia weight $w$ is one of the most important parameters in PSO; the global search ability of the algorithm will be improved with the help of the larger $w$ value, and a small $w$ value is to enhance the capacity of local optimization algorithm. According to different weights, $w$ can be divided into PSO linearly decreasing weights by adaptive weight method and random weight method [13].

Linearly Decreasing Weights [14]. Let inertia weight decrease linearly from the maximum value $w_{max}$ to $w_{min}$ at the beginning, a larger $w$ value is to optimum algorithm out of local conductively, and the latter algorithm is in favor of local space for precise search. Inertia weight $w$ relationship with the number is

$$
w = w_{max} - \frac{t \cdot \left( w_{max} - w_{min} \right)}{t_{max}}, \quad (12)
$$

where $w_{max}$ and $w_{min}$ denote the inertia weight maximum and minimum values, $t$ represents the current number of iterations, and $t_{max}$ is the maximum number of iterations.

Adaptive weight method is that the inertia weight $w$ with the fitness value of particles is automatically changed. This method takes into account the particle current fitness value $f$ and the relationship between the average fitness value $f_{aver}$ and the minimum fitness value $f_{min}$ in all particles. When the fitness value of all the particles tends to converge or be optimum, the inertia weight $w$ is greater; when the fitness value of all the particles scattered inertia weight $w$ takes a smaller value. Meanwhile, when the fitness value of particles is better than average fitness value $f_{aver}$, this corresponds to a smaller inertia weight; when the fitness value of particles is worse than average fitness value $f_{aver}$, this corresponds to
a larger inertia weight, so that the particles move closer to better search area. Inertia weight \( w \) is expressed as

\[
w = \begin{cases} 
    w_{\text{max}} - \frac{(w_{\text{max}} - w_{\text{min}}) \cdot (f - f_{\text{min}})}{(f_{\text{avg}} - f_{\text{min}})}, & f \leq f_{\text{avg}} \\
    w_{\text{max}}, & f > f_{\text{avg}}.
\end{cases}
\]

Random weight method [15] is that the inertia weight \( w \) obeys a certain random number distributed randomly. If, at the beginning of the algorithm, the particle position is close to the best point, linearly decreasing the weight of the larger \( w \) values may deviate from the optimum region, and random weights \( w \) may have a relatively small value, accelerating the convergence speed. If, at the beginning of the algorithm, the particles could not be found in the optimum area, the weights \( w \) method is decreased linearly because of diminishing, so, ultimately, the algorithm cannot be converged to the best advantage, and the random weight method can overcome this limitation. Therefore, in practical problems, some random weighting method can get better results than linear decline law. Inertia weight \( w \) is expressed as

\[
w = \mu + \sigma \cdot N(0, 1),
\]

\[
\mu = \mu_{\text{min}} + (\mu_{\text{max}} - \mu_{\text{min}}) \cdot \text{rand},
\]

wherein \( \mu \) represents a random weighted mean, \( \mu_{\text{max}} \) and \( \mu_{\text{min}} \), respectively, and the minimum and maximum random weights mean, \( \sigma \) represents a random weights variance, \( N(0, 1) \) represents the standard normal distribution of random numbers, and \( \text{rand} \) represents a random number between 0 and 1.

3.4. Particle Swarm Optimization Algorithm with Improved Learning Factor. In the practical application of the algorithm, the value of learning the way factor is \( c_1 = c_2 = 2 \); there are other variable learning factors; a common synchronous and asynchronous learning factor is changed.

Synchronous learning factor that is changed by \( c_1 \) and \( c_2 \) at the same time decreasing linearly, their relationship with \( t \) is as follows:

\[
c_1 = c_2 = c_{\text{max}} - \frac{c_{\text{max}} - c_{\text{min}}}{t_{\text{max}}} \cdot t,
\]

where \( c_{\text{max}} \) and \( c_{\text{min}} \) are the maximum and minimum learning factors; usually the maximum value is 2.1 and the minimum is 0.8.

Asynchronous learning factor changes [16] are \( c_1 \) and \( c_2 \) having various changes over time. Larger initial algorithm is \( c_1 \), \( c_2 \) is smaller, so that the particles have a greater self-learning ability and smaller social learning ability; the particles can search the entire search space globally. Later smaller algorithm \( c_1 \), \( c_2 \) has larger particles having a smaller self-learning ability and greater social learning ability; the particles can accurately search the optimum area. Learning factor is expressed in as

\[
c_1 = c_{\text{max}} - \frac{c_{\text{max}} - c_{\text{min}}}{t_{\text{max}}} \cdot t,
\]

\[
c_2 = c_{\text{min}} + \frac{c_{\text{max}} - c_{\text{min}}}{t_{\text{max}}} \cdot t.
\]

Ratnaweera et al. [17] found experimentally that, in most cases, \( c_{\text{max}} = 2.5 \), \( c_{\text{min}} = 0.5 \) can be taken to achieve the ideal solution.

3.5. Hybrid Particle Swarm Optimization. In addition to swarm intelligence algorithm and particle swarm optimization algorithm, but also including genetic algorithms, simulated annealing algorithm and firefly algorithm, each algorithm has its unique advantages. Hybrid particle swarm optimization refers to the other intelligent optimization algorithms into the ideological hybrid algorithm particle swarm optimization algorithm formation.

The genetic algorithm and particle swarm optimization algorithm combined GA-PSO algorithm is proposed by Premalatha and Natarajan [18]. The genetic algorithm of natural selection mechanism (Selection) applied to PSO; the basic idea is that in each iteration all the particles are sorted according to their fitness values and a good half of the particles are of fitness location and speed value, rather than another half that are sorted according to the position and velocity of a particle, while maintaining all particles fitness unchanged. By eliminating the difference between the particles, the algorithm can achieve faster convergence. Hybrid genetic algorithm mechanism (crossover) applied to PSO is that, in each iteration, randomly select a fixed number of particles into the hybrid cell, the particles cross the pool pairwise hybridization to give the same number of progeny particles with particle replacing the parent progeny particles iteration population. Wherein the position and velocity of the particle and offspring (18) is determined by formula (17):

\[
X_{\text{child}} = p \cdot X_{\text{parent1}} + (1 - p) \cdot X_{\text{parent2}},
\]

\[
V_{\text{child}} = v_{\text{parent1}} + v_{\text{parent2}} \cdot |v_{\text{parent1}}|,
\]

wherein \( p \) is a random number \([0, 1]\) and \( v_{\text{parent1}} \) or \( v_{\text{parent2}} \) can be chosen randomly as \( v_{\text{parent1}} \) or \( v_{\text{parent2}} \). By hybridization technology, it can improve particle swarm diversity, avoiding premature convergence algorithm.

Liu et al. [19] proposed the chaotic particle swarm optimization algorithm in order to optimize the particle swarm optimization algorithm. Chaos (chaos) is a nonlinear phenomenon in nature in a ubiquitous, periodicity, randomness, and intrinsic regularity. Periodicity of chaos embodied in it cannot be repeated through all the states in a search space; randomness is reflected in its performance similar to messy random variable, which embodies the inherent regularity in nonlinear systems under certain conditions defined in it. In addition, the chaotic initial conditions that are particularly sensitive to the initial value of the extremely weak changes
will cause a huge deviation in the system. Because chaos is easy to implement and make the algorithm out of local optima special properties, the researchers propose a chaotic optimization idea. Periodicity of chaos, randomness, and chaos inherent regularity of such thinking can be complementary optimization algorithm combined with PSO.

Victoire and Jeyakumar [20] proposed PSO and sequential quadratic programming (SQP) method for solving the combined economic dispatch (economic dispatch problem, EDP). SQP is a nonlinear programming method; it starts from a single point of search and uses gradient information obtained final solution. Research by three different EDP questions the validity of the method.

Lu et al. [21] have introduced the real value of the mutation operator (real-valued mutation, RVM) into the particle swarm optimization algorithm; the algorithm is used to improve the global search ability. Interestingly, when the RVM operator is applied to different functions, it can be operated effectively. By comparing the experiments, the authors found that a combination of shrinkage factor, inertia weight, and RVM operator mixed CBPSO-RVM algorithm can perform better in most of the test cases.

4. Particle Swarm Optimization Algorithm
Fused with Idea from Simulated Annealing

Particle swarm optimization algorithm can be easily trapped in the local optimum and result in premature convergence. Probabilistic jumping property of simulated annealing, SA, makes it possible to complementary associate with particle swarm optimization algorithm and fuses PSO global exploration capacity with SA local exploration capacity.

Annealing in metallurgy refers to heating and then cooling the material at a specific rate, which is for increasing the volume of crystal grains and reducing defects in the crystal lattice. At the beginning, material atom is at a position which has local minimum internal energy; then heating increases atoms energy, and atoms leave the initial position and move randomly to other locations. When annealing cools down, atoms are with low speed, so it is possible for atoms to find a location with lower internal energy than initial ones. Inspired by annealing of metals, researcher proposes simulated annealing to solve optimization problem. Optimization problem searches every potential solution to represent atoms location, and evaluation function for potential solution represents atoms internal energy at current location, wherein the optimization purpose is to find an optimization solution, hence getting a minimum value for evaluation function of this solution.

Simulated annealing is with mutation probability in searching process, which can effectively avoid the algorithm being trapped in the local optimum in iteration process. The key to this algorithm is to refuse local minimum solution in certain probability and then skip over local minimum point and continue to search other possible solutions in search area; also, the probability decreases along with temperature.

This paper fuses particle swarm optimization algorithm and simulated annealing to solve coverage restoring problem in wireless sensor actuator network. Algorithm detailed steps are as below.

Step 1. Initialize basic parameters, like population size, the maximum number of iterations, inertia weight, learning factor, annealing constant, and so on. Set upper as well as lower bounds for particle location and particle speed. Initialize each particle location in swarm as all nodes coordinate in network with coverage and randomly initialize each particle speed.

Step 2. Calculate fitness value $f(p_i)$ of each particle $p_i$. Take current location and fitness value of each particle as its history best location and fitness value $p_{best}$. Use location of particle $p_g$ with best fitness value as swarm history best location and corresponding best fitness value $f(p_g)$ as swarm best fitness value.

Step 3. Determine initial temperature according to the formula

$$T_0 = \frac{f(p_g)}{\ln 5}. \quad (19)$$

Step 4. According to fitness value $f(p_i)$ and global best value $f(p_g^g)$ of each particle $p_i$, calculate fitted value of each particle fitness value under current temperature:

$$TF(p_i) = \frac{\exp \left(\frac{f(p_i) - f(p_g^g)}{\tau}\right)}{\sum_{i=1}^{pop} \exp \left(\frac{f(p_i) - f(p_g^g)}{\tau}\right)}. \quad (20)$$

Step 5. According to fitted value of each particle fitness value, fuse with roulette strategy and confirm replacement value $p_g^g$ of global optimal particle $p_g$ from all particles. Then substitute fitted value into particle moving update equation to solve particle new speed and new location for applying in next iteration.

Step 6. Calculate fitness value of each particle and update particles $p_{best}$ and swarm $g_{best}$.

Step 7. Operate annealing according to formula (21), wherein $\lambda$ is for annealing constant and $k$ is for iterative times:

$$T_{k+1} = \lambda \cdot T_k. \quad (21)$$

Step 8. If the algorithm reaches either predicted operational precision or max iterative times, then algorithm ends. Or return to Step 4.

5. Simulation Experiment

5.1. Particle Parameter Description. Coverage restoring problem can be abstracted into nonconstrained optimization problem, which takes network coverage ratio as optimization target and nodes coordinates as decision variable. This chapter describes particle parameter. Particle location $X$ is for all nodes coordinates, which can be expressed as $X = \{x_1, y_1, x_2, y_2, \ldots, x_i, y_i, \ldots, x_N, y_N\}$, wherein $N$ is for nodes number and $x_i, y_i$ ($1 \leq i \leq N$) are for abscissa and ordinate of
In each algorithm, the nodes moving distance is controlled by particle upper \( V_{\text{max}} \) and lower \( V_{\text{min}} \) bounds. According to particle position, particle velocity can be expressed as fitness function value:

\[
\text{fitness} = \frac{1}{\eta} = \frac{A_g}{\sum_{i=1}^{A_s} P_g(G_i)}. \tag{22}
\]

5.2. Experiment Result and Analysis. In this paper, the solving method for coverage restoring problem is simulated in MATLAB 2012a as experimental environment. Nodes number \( N = 100 \), and monitor area side length \( L = 500 \) m, nodes sensing radius \( R = 30 \) m, nodes possibility sensing model parameter \( \rho = 0.5 \), and grids number \( A_g = 100 \). During initialization, nodes are deployed randomly and evenly in whole monitor area, and initial deployment of nodes is as shown in Figure 3. In this figure, every dot is for nodes, number beside is for node number, and circle region is for nodes sensing region. As shown in the figure, there are 4 obvious holes in initial deployment of nodes, and the solving target of network coverage problem is to move redundant nodes beside holes, hence increasing network coverage.

To verify effectiveness of particle swarm optimization algorithm which is based on simulated annealing, this paper uses basic particle swarm optimization algorithm and various improved algorithms to simulate and compare on network coverage problem. Figures 4–7 are the final deployments of nodes, which are simulated from basic particle swarm optimization algorithm, particle swarm optimization algorithm fused with crossover mutation idea from genetic algorithm, particle swarm optimization algorithm based on simulated annealing and with compression factor, and particle swarm optimization algorithm based on simulated annealing and using asynchronous learning factors. Figure 8 is comparison on best fitness value change in iteration process of each algorithm.

After analyzing simulation results from each algorithm, we can see that simulation effect from GPSO is the worst, and SAPSO with compression factor is the best. The final purpose of particle movement is to improve network coverage by moving redundant node and restoring network holes; at the meantime, to avoid too much energy consumption, moving distance of redundant node cannot be too far. In each algorithm, the nodes moving distance is controlled by particle upper \( V_{\text{max}} \) and lower \( V_{\text{min}} \) bounds. According to...
comparison experiment results, when particle velocity upper and lower values are taken from $v_{\text{max}} = 0.02 \cdot L$, $v_{\text{min}} = -v_{\text{max}}$, this achieves the best experiment effect, whereas if exceeding, network topology will change a lot, and if less, it will be hard to restore hole. Compared to other applications of PSO, in network coverage problem, the particle velocity must be set as small, so that the limitation will slow particle location change, hence decreasing variety of particle swarm; therefore, it will be hard to improve PSO performance by using crossover mutation from genetic algorithm, and experiment shows that GPSO effect is even worse than BPSO effect. Both SAPSO with compression factor and SAPSO with asynchronous learning factors have good simulation effect. As shown in $g_{\text{best}}$ variation curve, algorithm can skip out local optimum constantly to find better particle location. As shown in final deployment of nodes, the big holes among nodes sensing circle almost disappear, but there are still small holes; however, considering intruder mobility in monitor area, the intruder will inevitably enter nodes sensing region, so small holes can be ignored.

6. Conclusion

As the wireless sensor actuator network usually work in poor environment, like battlefield, fire, and so forth, it is most likely to exhaust energy, suffer irresistible damage, or cause network coverage hole due to the long moving distance. This paper proposes a coverage restoring method by moving nodes besides holes areas and transforming coverage restoring problem into nonconstrained optimization problem which takes network coverage ratio as optimization target. As it is hard to get analytical solution for this optimization problem, swarm intelligence algorithm is needed to do random iterative search. After comparison simulation results from BPSO, GPSO, and SAPSO with nonconstrained optimization problem, it verifies that simulated annealing can well combine with particle swarm optimization algorithm to fulfill algorithm early global search and later local detection. Simulation proves that hybrid algorithm can effectively solve hole coverage problem in wireless sensor actuator network.

Competing Interests

The authors declare that they have no competing interests.

References


Submit your manuscripts at http://www.hindawi.com