Hindawi
Journal of Sensors
Volume 2021, Article ID 5582646, 12 pages
https://doi.org/10.1155/2021/5582646

Research Article

An Improved Adaptive Clone Genetic Algorithm for Task Allocation Optimization in ITWSNs

Zhihua Zha,[1] Chaoqun Li,[2] Jing Xiao,[2] Yao Zhang,[3] Hu Qin,[2] Yang Liu,[2] Jie Zhou,[2] and Jie Wu[1]

[1]*Mechanical and Electrical Engineering, Shihezi University, Shihezi 832000, China*
[2]*College of Information Science and Technology, Shihezi University, Shihezi 832000, China*
[3]*University of the Cordilleras, Baguio City 2600, Philippines*

Correspondence should be addressed to Jie Wu; wjshz@126.com

Received 18 February 2021; Revised 13 March 2021; Accepted 17 March 2021; Published 7 April 2021

Academic Editor: Bin Gao

Research on intelligent transportation wireless sensor networks (ITWSNs) plays a very important role in an intelligent transportation system. ITWSNs deploy high-yield and low-energy-consumption traffic remote sensing sensor nodes with complex traffic parameter coordination on both sides of the road and use the self-organizing capabilities of each node to automatically establish the entire network. In the large-scale self-organization process, the importance of tasks undertaken by each node is different. It is not difficult to prove that the task allocation of traffic remote sensing sensors is an NP-hard problem, and an efficient task allocation strategy is necessary for the ITWSNs. This paper proposes an improved adaptive clone genetic algorithm (IACGA) to solve the problem of task allocation in ITWSNs. The algorithm uses a clonal expansion operator to speed up the convergence rate and uses an adaptive operator to improve the global search capability. To verify the performance of the IACGA for task allocation optimization in ITWSNs, the algorithm is compared with the elite genetic algorithm (EGA), the simulated annealing (SA), and the shuffled frog leaping algorithm (SFLA). The simulation results show that the execution performance of the IACGA is higher than EGA, SA, and SFLA. Moreover, the convergence speed of the IACGA is faster. In addition, the revenue of ITWSNs using IACGA is higher than those of EGA, SA, and SFLA. Therefore, the proposed algorithm can effectively improve the revenue of the entire ITWSN system.

1. Introduction

Nowadays, with the rapid increase of vehicles, the phenomenon of traffic congestion and pollution is getting worse, which leads to frequent traffic violations and accidents. These have become bottlenecks for the further development of cities [1]. Therefore, it is urgent for the transportation department to apply a more advanced and intelligent data acquisition means to obtain the massive data of the transportation industry. Transportation departments can provide real-time and accurate traffic information services for passengers and provide a reference for staff to deal with emergencies and traffic violations [2, 3], for example, detect vehicles at intersections in all directions to improve simplification, signal control algorithms, and traffic efficiency based on the monitoring results. At the same time, the growth of existing roads and other hardware facilities can no longer satisfy the ever-increasing traffic problems, and intelligent transportation systems are getting more and more attention [4, 5].

An intelligent transportation system mainly includes the collection, transmission, control, and guidance of traffic information. A wireless sensor network can provide an effective means for information collection and transmission of the intelligent transportation system. The effective operation of the intelligent transportation system depends on obtaining comprehensive, accurate, and real-time dynamic traffic information. People process the information collected by sensor nodes to obtain comprehensive traffic condition, which facilitates identification, decision-making, positioning, detection, and tracking of vehicles in the traffic management. Therefore, information collection has become a hot issue in the development of intelligent transportation [6]. The traffic

conditions of different road sections are also different, and the tasks assigned by the sensors are also different. To better detect the traffic condition in the case of limited traffic sensor resources, an efficient task allocation strategy can collect traffic to a greater extent information for monitoring traffic conditions more effectively [7–9]. The focus of the research is to develop new task allocation schemes to maximize the network throughput and revenue of intelligent transportation wireless sensor networks (ITWSNs).

In the whole development process of ITWSNs, efficient QoS task allocation strategies have played an irreplaceable important role and have been widely used in various fields of transportation [10–12]. Scholars have extensively studied the task allocation problem in wireless sensor networks [13–15]. Paper [16] finds the key subtasks based on the estimated completion time of the subtasks and the weight coefficients and preferentially selects node assignments with strong capabilities and high processing efficiency. Paper [17] mixes the adaptability of particle swarm optimization with the flexible ability of dynamic alliance and obtains the fitness value through the weighting method to obtain the global optimal allocation method. Paper [18] allocates tasks to different clusters to achieve the goal of high benefit and then allocates tasks from the clusters to appropriate sensor nodes to balance the energy loss of the network. Paper [19] proposes a dynamic joint task allocation algorithm using linear programming to obtain a more balanced task allocation strategy. However, none of the above methods can better improve the revenue of task allocation in ITWSNs.

Aiming at the problem of maximizing revenue, the task allocation optimization technology is applied to ITWSNs, and it can greatly improve the overall revenue of the network. The main contributions are as follows:

(1) Firstly, this paper proposes an improved adaptive clone genetic algorithm (IACGA) to solve the optimization problem of task allocation, designs the task allocation model of ITWSNs, and designs a new fitness function to evaluate the performance of the algorithm

(2) Secondly, new adaptive operators and clone operators are designed to improve the optimization ability of the algorithm. IACGA combines adaptive operator and clonal operator, which has better performance, enhances global search ability, and avoids falling into local optimum

(3) Finally, the simulation results of IACGA, EGA, SA, and SFLA in ITWSN task allocation are compared to verify the superiority of IACGA in task allocation optimization, and the detailed data and discussion are given

The remaining structure of the paper is shown below. Section 2 introduces related research in the field of task allocation in intelligent transportation. Section 3 shows the task allocation model. Section 4 proposes an improved adaptive clone genetic algorithm to solve the problem of task allocation in ITWSNs. Section 5 illustrates the effectiveness of IACGA in solving the task allocation problem through simulation experiments and discusses it. Section 6 is the conclusion part.

2. Related Work

Task allocation is a classic problem widely studied in the field of ITWSNs, and its application in the field of wireless sensor networks in the intelligent transportation system is also crucial. However, wireless sensor network resources are severely limited, and existing algorithms cannot be directly applied. Paper [20] proposed a nested optimization technology based on a genetic algorithm to perform energy-efficient task allocation in a multihop cluster network. Generalized optimization goals can not only meet the real-time requirements of the application but also achieve energy efficiency. The optimization solution is obtained by combining the processes of task mapping, routing path allocation, and task scheduling based on genetic algorithms. The task graph simulation experiment is randomly generated, and the results show that the nested optimization technology has better performance than the random optimization technology. However, due to the high complexity of the algorithm, the efficiency of the program is not good.

In paper [21], for the task allocation problem of the urban road traffic information collection sensor network for the collaborative collection of complex traffic parameters, the sensor network is mapped to a multiagent system, and the task completion time, node energy consumption, and network load balance are used as the evaluation function. The authors used alliance-based collaborative methods to construct a nonlinear multiobjective optimization model of sensor network task allocation. The authors used genetic simulated annealing to search for the optimal alliance structure to achieve task allocation strategy optimization. Simulation experiments are carried out in the actual scene of road traffic information collection; the results show that the genetic simulated annealing can effectively optimize the alliance structure of task allocation. Compared with other optimization algorithms, the optimized model has low fitness function value, short task completion time, and low network energy consumption. This method can be used for traffic-oriented collaborative detection task allocation problems of the information collection sensor network. But the algorithm has slow convergence speed and poor performance.

Paper [22] proposed a hierarchical optimization task scheduling algorithm based on the characteristics of wireless sensor network task scheduling subject to deadlines and node energy constraints. In the paper, tasks are prioritized according to the threshold established by the deadline. Tasks with more urgent deadlines are assigned first to improve the success rate of scheduling. For tasks with loose deadlines, the goal is to reduce energy consumption and balance the load to increase the network revenue. Simulation experiments show that the algorithm has achieved better results in improving the success rate of task scheduling and balancing network load. However, in the case of limited computing power, the computational complexity of the algorithm increases exponentially with the increase in the number of network nodes.

Regarding the task allocation algorithm of intelligent transportation, there is a lot of progress in research. However, the existing research rarely involves complex perception tasks that require multiperson collaboration in group intelligence perception. Paper [23] studies this type of task. First, a

location-related task allocation problem for collaborative group intelligence perception is presented, and a formal analysis is carried out on it; then, it is proved that the problem is NP-hard to solve, and a greedy strategy and strategy based on this problem are proposed. However, this algorithm has high complexity and poor performance.

In the paper [24], the authors used the SFLA to explore the task allocation problem. Under the premise of task allocation modeling, the improved SFLA is used to solve the model. First, according to the characteristics of the target, the model solution matrix is designed by using a decimal encoding method. From this matrix, the sensor decision matrix can be directly obtained. On the basis of the original SFLA, by introducing a variable step size related to the number of iterations, the execution process of the algorithm is converted from a multipoint mutation mode to a single-point mutation mode, so that the algorithm has stronger robustness. Under the premise of satisfying the task allocation constraints, specific solutions and steps are given according to the principle of the algorithm. The feasibility and effectiveness of the algorithm are verified by example simulation. However, the algorithm is prone to premature convergence.

In the paper [25], the authors established a mathematical model of the road condition target allocation problem through the analysis of the factors affecting the target allocation in the road, and then, based on the idea of a hybrid optimization algorithm, combined the heuristic search mechanism with the simulated annealing, and proposed an improved greedy simulated annealing algorithm. The simulation results show that the algorithm is effective and feasible and can give a better optimal allocation plan. However, it is still easy to fall into a local optimal solution.

In view of the problems existing in the above literature, we have proposed a new solution to solve the problem of task allocation optimization and maximize the overall benefits of the ITWSN system, thereby reducing costs and improving benefits for the collection of information for intelligent transportation systems.

3. System Model

In a complex traffic information collection environment, in order to maximize the network benefits of ITWSNs, a mathematical model of ITWSNs is designed for the constraints of control range and computing power.

This paper proposes a task allocation model for ITWSNs located in the coordinates of the traffic area. This model can realize the continuous coordination of heterogeneous agents between groups under communication constraints, thereby maximizing the network revenue of wireless sensor networks. This model can be simply abstracted into N tasks and S traffic remote sensing sensor nodes in ITWSNs. Suppose S sensor nodes are randomly installed in the road traffic area to perform N tasks of road traffic information collection. The goal of this paper is to obtain the maximum network revenue value by assigning different tasks on different sensor nodes.

It is assumed that the advantages of each task are evaluated before task assignment, and the urgency of task assignment is estimated. In formula (1), the urgency of the n_{th} task assigned to the sensor can be represented by u_n, and the advantage of the n_{th} task assigned to the s_{th} sensor can be represented by $p_{s,n}$. From this, the revenue value of the sensor node performing the task can be represented by r_n. In formula (2), R represents the income of ITWSN task distribution, and it is expected to obtain the maximum revenue max (R).

$$r_{s,n} = u_n p_{s,n}, \tag{1}$$

$$R = \sum_{n=1}^{N} r_{s,n}, \tag{2}$$

$$P_{s,n} = \begin{bmatrix} p_{11} & p_{12} & p_{13} & p_{14} & p_{15} \\ p_{21} & p_{22} & p_{23} & p_{24} & p_{25} \\ p_{31} & p_{32} & p_{33} & p_{34} & p_{35} \end{bmatrix} (s \in [1, 3], n \in [1, 5]). \tag{3}$$

In formula (4), U_n represents the urgency of each task:

$$U_n = [u_1\ u_2\ u_3\ u_4\ u_5]\ (n \in [1, 5]). \tag{4}$$

At this time, suppose there are 1 individual, 5 tasks, and 3 sensor nodes in the model. Individuals naturally generate a coding solution randomly in formula (5), and D represents a task allocation solution. Randomly generate a task allocation solution, and the result is expressed in formula (5), and the corresponding coding method of the allocation plan is given in Section 4:

$$D = [3 \quad 2 \quad 2 \quad 1 \quad 3]. \tag{5}$$

In order to speed up the execution efficiency, the next step here is to convert the real number encoding to the binary encoding method. The binary matrix is shown in

$$D^* = \begin{bmatrix} 0 & 0 & 0 & 1 & 0 \\ 0 & 1 & 1 & 0 & 0 \\ 1 & 0 & 0 & 0 & 1 \end{bmatrix}, \tag{6}$$

$$P^*_{s,n} = P_{s,n} D^*, \tag{7}$$

$$P^*_{s,n} = \begin{bmatrix} 0 & 0 & 0 & p_{14} & 0 \\ 0 & p_{22} & p_{23} & 0 & 0 \\ p_{31} & 0 & 0 & 0 & p_{35} \end{bmatrix} (s \in [1, 3], n \in [1, 5]). \tag{8}$$

According to (3) and (6), $P^*_{s,n}$ is calculated in (7), which represents the advantage of assigning the n_{th} task to the s_{th} sensor. The result matrix is shown in formula (8). D^* is a binary task allocation plan matrix. $D^*_{s,n} = 0$ means that the f_{th} task is not allocated to the e_{th} sensor. $D^*_{s,n} = 1$ means that the n_{th} task is assigned to the s_{th} sensor. $P_{s,n}$ is a task allocation advantage matrix, where the value represents the advantage value of the n_{th} task assigned to the s_{th} sensor.

The identity matrix with column s in P_n is generated according to formulas (9) and (10). The number of sensors in the model corresponds to the number of columns in b_s. The calculated P_n is shown in formula (11).

$$b_s = [1 \quad 1 \quad 1], \tag{9}$$

$$P_n = b_s \cdot P^*_{s,n}, \tag{10}$$

$$P_n = [p_{31} \quad p_{22} \quad p_{23} \quad p_{14} \quad p_{35}]. \tag{11}$$

In formula (12), R_n represents the total benefit value of a group of task allocation plans:

$$R_n = P_n \cdot U'_n. \tag{12}$$

In the process of task allocation, due to the computing power and QoS constraints of sensor nodes, the two factors will affect the network revenue in the task allocation model. Bandwidth functions ($B(e)$), delay functions ($D(e)$), delay jitter functions ($D^{-}J(e)$), and loss packet rate functions ($P^{-}L(e)$) are designed. Suppose that there are four factors in a link v_i and v_j in ITWSNs. The QoS constraints are expressed in

$$B(l(v_i, v_j)) = \min\{B(e)\}, \tag{13}$$

$$D(l(v_i, v_j)) = \sum_{e \in l(v_i, v_j)} D(e), \tag{14}$$

$$D\_J(l(v_i, v_j)) = \sum_{e \in l(v_i, v_j)} D\_J(e), \tag{15}$$

$$P\_L(l(v_i, v_j)) = 1 - \prod_{e \in l(v_i, v_j)} (1 - P\_L(e)), \tag{16}$$

where e represents the link transmission energy, $l(v_i, v_j)$ represents a link, $B(l(v_i, v_j))$ represents minimum bandwidth, $D(l(v_i, v_j))$ represents total jitter, $D^{-}J(l(v_i, v_j))$ represents total delay jitter, and $P^{-}L(l(v_i, v_j))$ represents the total packet loss rate.

In ITWSNs, the task allocation of sensor nodes must not only meet actual traffic control requirements but also improve the QoS and revenue of the entire network. The higher the efficiency of task execution, the higher the overall profit of the system. System benefit refers to the sum of the benefits of each sensor node to complete the task. Because the traffic remote sensing sensor nodes in ITWSNs are heterogeneous, the assigned execution efficiency of the nodes is different, so the total task revenue of the sensor nodes may also be different. An excellent task distribution scheme can usually achieve better overall network service quality and higher overall network efficiency. Therefore, the allocation plan needs to consider the urgency and load size of each task and the execution capability of each sensor node to determine a task scheduling strategy.

4. IACGA for Task Allocation Optimization in ITWSNs

Aiming at the task allocation problem in ITWSNs, an optimization algorithm based on IACGA is proposed. This idea comes from biological evolution in nature. In our IACGA strategy, new adaptive strategies and cloning strategies were designed. These strategies enable IACGA to allocate tasks well and quickly find the best solution.

Traditional genetic algorithms tend to converge prematurely and fall into the local optimum. Therefore, in the IACGA we proposed, an adaptive mechanism is designed. In the crossover and mutation stage, the probability is adjusted according to the current algorithm operation, thereby affecting the global search capability of the entire algorithm. We have added a clonal immune mechanism. After the population fitness evaluation is completed, we will find the best individual to clone and then immunize to form the next generation of population. The resulting new population has the advantages of diversity and close to the optimal solution. Due to the complex traffic conditions and the task conditions of various heterogeneous sensors, the use of a clonal immune mechanism can effectively improve the efficiency of the algorithm, better solve the task allocation problem, and increase the system revenue of ITWSNs.

The main execution steps of IACGA are shown in Figure 1.

We can use the following detailed steps to illustrate the algorithm flow shown in Figure 1.

Step 1. Initialize the population. First, construct a parent population that satisfies the conditions of the model. The population can be abstracted as a real matrix. Suppose there are M individuals and N tasks and the size of the matrix is MN.

Step 2. Calculate the fitness of the parent and perform selection operations. Sort according to the fitness of each individual to find the best individual.

Step 3. Clone the best individual. Clone the most adaptable individual, use immunity to decide whether to accept, and recombine a new population.

Step 4. Adjust parameters adaptively. Judge whether the population fitness is clustered, and adjust the parameters of crossover and mutation.

Step 5. Perform crossover and mutation according to the parameters in Step 4.

Step 6. Repeat Steps 2–5, and reach the maximum number of iterations to meet the termination condition.

Step 7. Terminate the algorithm and output the best task allocation plan.

In Algorithm 1, we show the entire IACGA algorithm flow in pseudocode.

This section discusses several parts of IACGA from the aspects of task allocation coding scheme and population

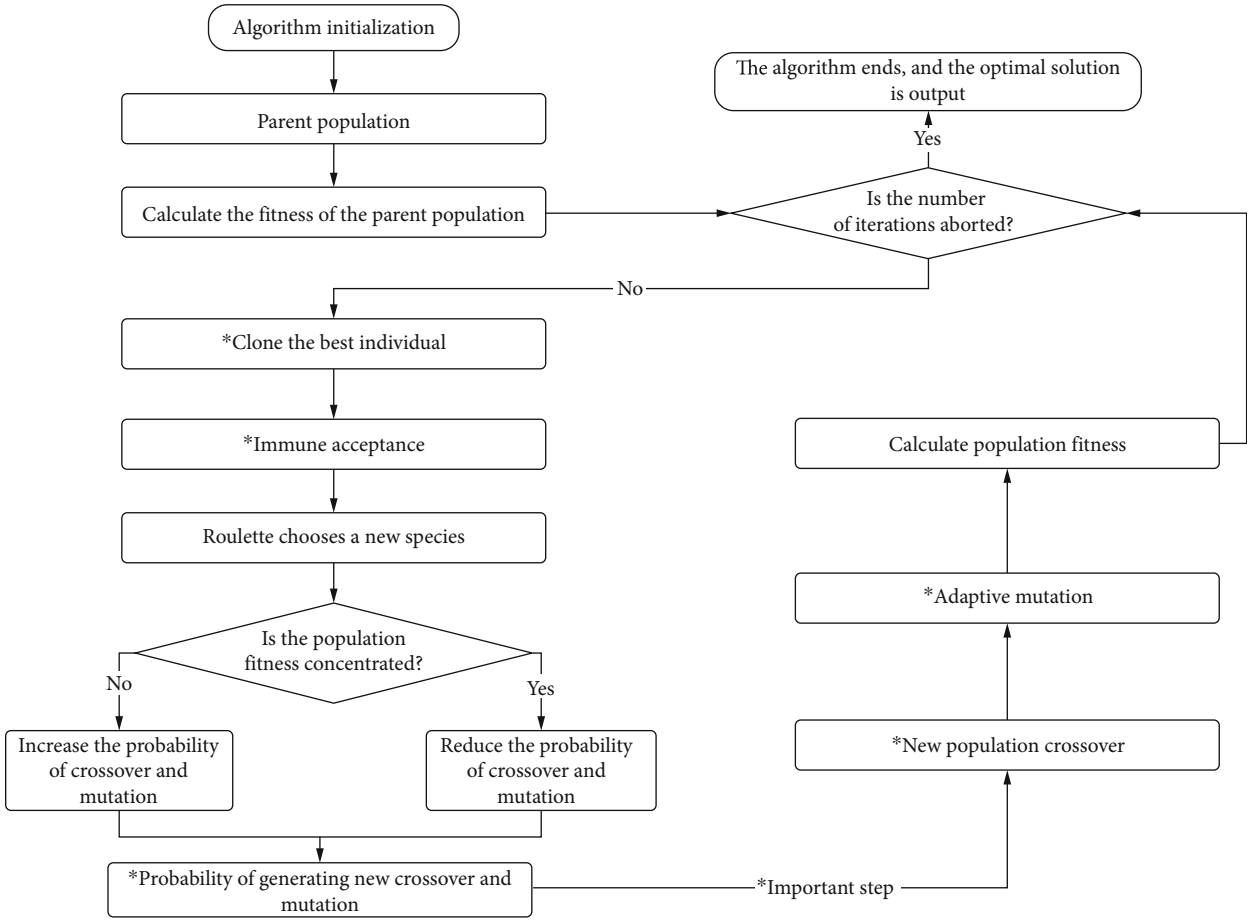

Figure 1: The flow chart of IACGA.

```
IACGA for Task Allocation Optimization in ITWSNs
  BEGIN
    Initialize the population, using random methods;
    Set generations = maximum iteration;
    For generation =1: generations
      Calculate the fitness of the parent;
      Find the individuals with the highest fitness;
      Clone the best individual;
      Immune acceptance;
      if The population fitness converge
        Increase the probability of crossover and mutation;
          else
            Reduce the probability of crossover and mutation;
      end
      Crossover operation;
      Mutation operation;
    end For
    Output the best task allocation scheme;
  END
```

Algorithm 1: IACGA's algorithm pseudocode.

initialization, fitness calculation, selection, crossover, mutation, and optimization operators.

4.1. Coding Scheme. Coding is the first important step to solve the task allocation problem. The task allocation problem of the sensor network is to allocate different tasks to different sensors, so as to get the greatest benefit. The coding idea is to treat a set of allocation schemes as a chromosome with multiple genes. The coding method directly affects the running of the program, the calculation of fitness, and subsequent crossover and mutation operations. Therefore, this article uses integer encoding to facilitate the execution of the algorithm and improve the readability of the program. Suppose there are M individuals in the population, S nodes in the sensor network, and N tasks to be assigned. In formula (17), $d_{m,n}$ represents the situation where the n_{th} task in the m_{th} individual is allocated to the s_{th} sensor:

$$D_{(M,N)} = \begin{bmatrix} d_{1,1} & d_{1,2} & \cdots & d_{1,N-1} & d_{1,N} \\ d_{2,1} & d_{2,2} & \cdots & d_{2,N-1} & d_{2,N} \\ \cdots & \cdots & d_{m,n} & \cdots & \cdots \\ d_{M-1,1} & d_{M-1,2} & \cdots & d_{M-1,N-1} & d_{M-1,N} \\ d_{M,1} & d_{M,2} & \cdots & d_{M,N-1} & d_{M,N} \end{bmatrix} \cdot (d_{m,n} \in [1, S], m \in [1, M], n \in [1, N]). \tag{17}$$

The allocation scheme of 4 chromosomes, 8 sensors, and 9 tasks is expressed in

$$D_{(4,9)} = \begin{bmatrix} 1 & 7 & 7 & 3 & 5 & 8 & 2 & 6 & 1 \\ 4 & 2 & 6 & 8 & 8 & 5 & 3 & 3 & 7 \\ 8 & 3 & 1 & 1 & 6 & 3 & 5 & 5 & 4 \\ 2 & 1 & 6 & 5 & 5 & 2 & 7 & 8 & 3 \end{bmatrix}. \tag{18}$$

4.2. Initial Population. The population is coded according to the task allocation model. Its purpose is to establish an abstract connection between task assignment and IACGA. The population initialization can adopt a random scheme to better simulate the natural environment. Therefore, M natural biological individuals are generated in the initialized population. The population can be simply described as $D = \{D_1, D_2, \cdots, D_M\}$. The m_{th} biological individuals can be expressed as $D_m = \{d_{m,1}, d_{m,2}, \cdots, d_{m,N}\}$. The specific gene coding example is shown in formula (18).

4.3. Fitness Evaluation. The fitness function determines the convergence speed of IACGA to a certain extent. Each individual in the population has its own fitness value. IACGA embodies the process of biological evolution in nature, selecting good individuals through fitness and weeding out poor individuals. In this study, individuals are evaluated based on the value of income from task allocation, that is, fitness value. The goal of the research is to maximize network revenue. The profit value of task allocation can be calculated by formulas (1) and (2).

4.4. Selection. The fitness of each individual in the initialized population is very different. In order to speed up the algorithm solving speed, IACGA finds the highest individual through the fitness of each individual, that is, the optimal individual. Our goal is to make highly adaptable individuals inherit as much as possible into the next generation population. The environmental adaptability of the new population will become stronger. The idea of selection operation comes from Darwin's theory of biological evolution. A reasonable selection method can enhance the optimization ability of the algorithm. This paper uses the roulette selection operator to calculate the relative fitness value of all individuals in the population according to formula (2) and then calculates the probability of each individual being selected according to formula (19). We assume that there are M individuals in the population:

$$P(m) = \frac{R(m)}{\sum_{m=1}^{M} R(m)} \ (m \in [1, M]), \tag{19}$$

where $P(m)$ represents the probability of the m_{th} individual being selected, $R(m)$ represents the fitness value of the m_{th} individual, and $\sum_{m=1}^{M} R(m)$ represents the total fitness of the population.

4.5. Crossover. The essence of crossover operation is the exchange of partial structures of two individuals. After the parent population crosses, new offspring populations are produced, which increases genetic diversity. The crossover operation not only guarantees the stable evolution of the population but also makes the algorithm develop in the direction of the optimal solution. The probability of crossover also greatly affects the convergence speed of the algorithm. This paper proposes the strategy of adaptively adjusting the cross probability to ensure the global search ability of the algorithm. At the same time, the single-point crossover method is adopted, and random factors are added to find the crossover point and exchange structure.

In order to better understand the crossover process, we use formulas (20) and (21) to visualize it. We chose the crossover point at the 5_{th} gene sequence and swapped all sequences from then on:

$$D_1 = \begin{bmatrix} 1 & 7 & 7 & 3 & | & 5 & 8 & 2 & 6 & 1 \\ 4 & 2 & 6 & 8 & | & 8 & 5 & 3 & 3 & 7 \end{bmatrix} \text{(before)}, \tag{20}$$

$$D_2 = \begin{bmatrix} 1 & 7 & 7 & 3 & | & 8 & 5 & 3 & 3 & 7 \\ 4 & 2 & 6 & 8 & | & 5 & 8 & 2 & 6 & 1 \end{bmatrix} \text{(after)}, \tag{21}$$

where D_1 represents the gene sequence of the two chromosomes before the crossover representing the task allocation plan and D_2 represents the gene sequence after the two chromosomes cross.

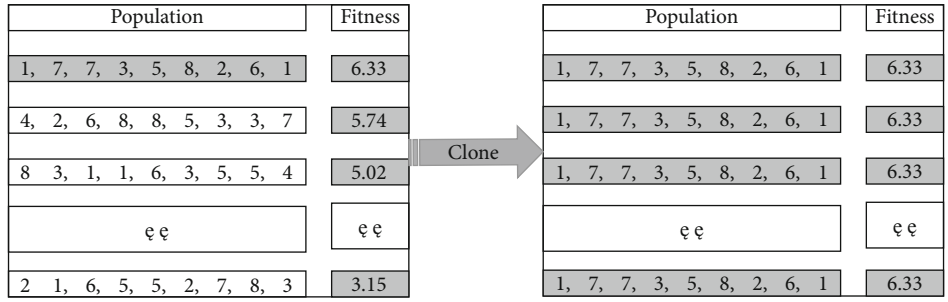

Figure 2: The process of cloning elite individuals.

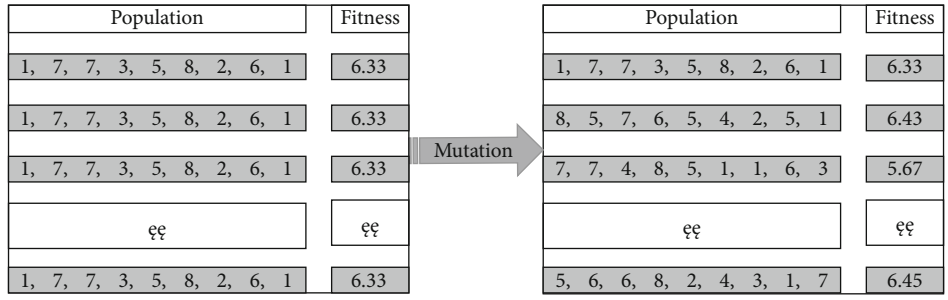

Figure 3: The process of population variation.

4.6. Mutation. Mutation operation is a crucial operation in genetic evolution. Mutation causes random changes in certain parts of some individuals in the population, resulting in new genes. Compared with the crossover operation, mutation is an auxiliary operation, and mutation is used to mutate genes on a single chromosome. Mutation also increases the genetic diversity of the population, increases the search space, and avoids premature convergence of the algorithm. The mutation probability used in the mutation process of traditional genetic algorithms is usually fixed. If the setting is too large, the population structure is unstable, and the optimal solution cannot be found. If the setting is too low, the evolution of the population will be stagnated, falling into a local optimal solution, and the algorithm will converge prematurely. Therefore, this study designed a new adaptive mutation factor in the mutation and adjusted the mutation probability in real time according to the degree of fitness aggregation of the algorithm, so as to achieve the purpose of optimizing task allocation. In order to better understand the mutation process, we use formulas (22) and (23) to visualize it. We selected the mutation points at the 3_{rd} and 7_{th} gene sequences.

$$D_3 = [1 \quad 7 \quad *7 \quad 3 \quad 5 \quad 8 \quad *2 \quad 6 \quad 1] \text{(before)}, \tag{22}$$

$$D_4 = [1 \quad 7 \quad *2 \quad 3 \quad 5 \quad 8 \quad *4 \quad 6 \quad 1] \text{(after)}, \tag{23}$$

where D_3 represents the gene sequence before mutation of a chromosome representing the task allocation plan and D_4 shows the gene sequence after a chromosome mutation.

4.7. Adaptive Operator. The adaptive mechanism designed in this study is mainly used in the crossover and mutation stages. Crossover and mutation will change the original genes of individuals and are the main means to ensure population diversity. In the past classic biological evolution algorithms, scholars usually set some constants, and the constants remain unchanged during the entire algorithm running cycle. This can reduce the difficulty of algorithm design, but the final result of the algorithm is often not good. We set the probability of crossover and mutation in the algorithm as PC and PM, respectively. If the two are set larger, it will affect the stability of the algorithm, and the smaller setting will affect the evolution speed of the algorithm. Therefore, we designed an adaptive strategy to change PC and PM in real time according to the fitness value. In order to prevent PC and PM from approaching 0 and the algorithm from falling into the local optimum, we designed formulas (24) and (25), with the size of the individual fitness value changes nonlinearly to quantify the probability of crossover and mutation:

Table 1: Key experimental parameters of IACGA.

Algorithm	Number of generations	Population size	Adaptive crossover probability	Adaptive mutation probability
IACGA	100	50	0.65~0.88	0.01~0.04

$$PC(m) = PC_{\min} + \frac{PC_{\max} - PC_{\min}}{1 + \exp\left[k \cdot R(m) - R_{\text{avg}}/R_{\max} - R_{\text{avg}}\right]}, \tag{24}$$

$$PM(m) = PM_{\min} + \frac{PM_{\max} - PM_{\min}}{1 + \exp\left[k \cdot R(m) - R_{\text{avg}}/R_{\max} - R_{\text{avg}}\right]}, \tag{25}$$

Table 2: Key experimental parameters of EGA.

Algorithm	Number of generations	Population size	Crossover probability	Mutation probability	Percentage of elites
IACGA	100	50	0.75	0.04	10%

where PC_{max} and PC_{min} are the upper limit of 0.88 and the lower limit of 0.65 for probability adjustment, respectively, k is set to 15, and PM_{max} and PM_{min} are the upper limit of 0.04 and the lower limit of 0.01 for mutation probability adjustment, respectively. R_{max} is the maximum value of fitness of individuals in the population, and R_{avg} is the average fitness value of all individuals in the population, and $R(m)$ represents the fitness value of the m_{th} individual.

4.8. Clone Operator. In order to improve the convergence speed and diversity of the algorithm, we propose a cloning mechanism to be applied to the selection and mutation stages. The advantage of the cloning mechanism is that clonal amplification can increase the convergence speed of optimization calculations, while clonal mutation can maintain the diversity of the population. The main operation of cloning is asexual reproduction, and the performance mainly depends on the quality of the clonal mutation operation. Choose a small mutation probability for operation. At this time, the local search of the algorithm is very fine, but the global search ability of the algorithm is poor, and it is easy to fall into the local optimal solution, resulting in a waste of computing resources. The key of the artificial immune system to solve the problem is to use the principle of immunodominance in immunology to continuously change the antibody itself in response to the stimulation of the antigen. Immune memory cloning basically runs in parallel on two groups, namely, antibody population and memory unit, thus more comprehensively simulating the clonal selection process of the biological immune system. The clonal expansion in this study adopts the elite cloning model and clones the elite individuals with the greatest population fitness. Figures 2 and 3 clearly show the process of clonal expansion and mutation, respectively. The clone mutation operation adopts the adaptive mutation mechanism designed by us. The mutation process is similar to that shown in formulas (24) and (25).

5. Simulation and Discussion

5.1. Experimental Setup. In this section, we will use simulation to test the algorithm performance of IACGA in ITWSN task allocation and compare the simulation results with EGA, SA, and SFLA in ITWSN task allocation. The results of the four algorithms are the average of 100 experiments. Under the same other conditions, different sensor nodes and tasks of different scales are used for simulation comparison in ITWSNs. The hardware computing environment is a PC with Intel® Core™ i5 2.30 GHz CPU. The software computing environment is the Windows 10 operating system of the same version number, MATLAB R2018a.

In this simulation, in order to better compare the performance of IACGA with EGA, SA, and SFLA, we set the population size of the four algorithms of IACGA, EGA, SA, and SFLA to 50, and the maximum number of generations to 100. In IACGA, we set the probability range of adaptive crossover to 0.65~0.88 and set the probability range of adaptive mutation to 0.01~0.04. In EGA, we set the probability of crossover to 0.75, set the probability of mutation to 0.04, and set the percentage of elites to 10%. In SA, we set the initial temperature to 200 and the annealing factor to 0.83. In SFLA, the grouping of frogs is set to 5, the number of frogs in each group is 10, and the maximum step length of frog jumping is set to 1. For IACGA, EGA, SA, and SFLA, the experimental parameters of each algorithm are shown in Tables 1–4, respectively.

Table 3: Key experimental parameters of SA.

Algorithm	Number of generations	Population size	The initial temperature	Annealing factor
SA	100	50	200	0.83

Table 4: Key experimental parameters of SFLA.

Algorithm	Number of generations	Population size	Frog group	Frogs in each group	Maximum step size
SFLA	100	50	5	10	1

5.2. Discussion of Experimental Results. In the part of discussion of experimental results, the simulation data will be analyzed in many aspects. The results of the experiment are shown in the form of a simulation evolution curve, bar chart, and data table and analyzed and discussed.

Figures 4(a)–4(d) intuitively give the simulation results of IACGA, EGA, SA, and SFLA in four different numbers of sensor nodes and tasks. On the whole, in the four different scale situations, the optimization performance of IACGA is better than EGA, SA, and SFLA. EGA performance is better than that of SA and SFLA. It can be seen from Figures 4(a)–4(d) that when the algorithm runs to the 50th generation, IACGA has basically converged and reached a good solution. At this time, IACGA's network revenue is far greater than EGA, SA, and SFLA. SA has converged and fell into a local optimal solution in 30 generations. The convergence speed of SFLA is slow, and the degree of optimization is much lower than that of IACGA, EGA, and SA. Especially in Figure 4(d), when the sensor node is 50 and the task is 120, the network revenue of IACGA increases by 44.0117; EGA, SA, and SFLA are, respectively, 42.7619, 41.7644, and 40.8958. In general, under the conditions of running 150 generations, by proposing adaptive strategies and cloning strategies, IACGA can well solve the problem of revenue from task allocation and has better speed and performance than EGA, SA, and SFLA.

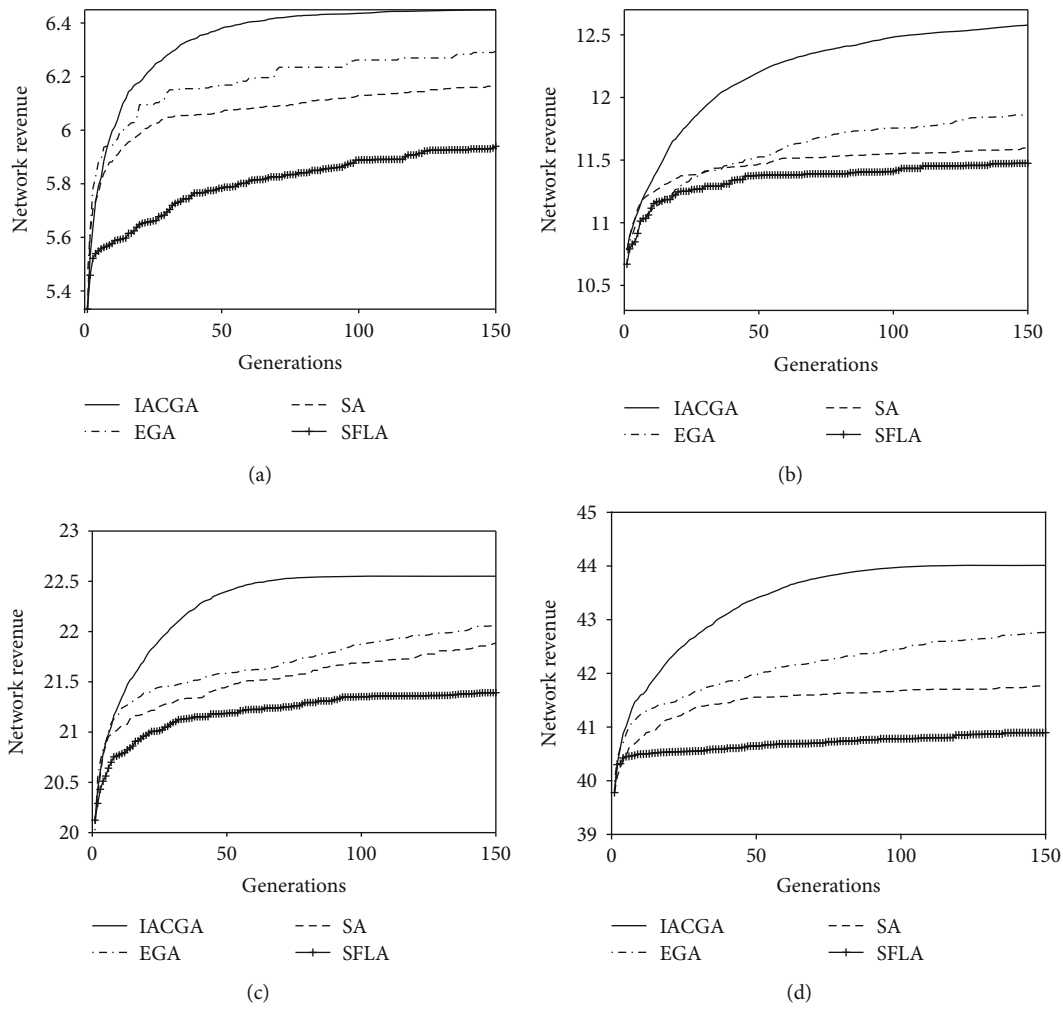

Figure 4: Comparison of network revenue optimization trends of four algorithms: (a) the result of running 150 generations of 15 tasks and 8 nodes; (b) the result of running 150 generations of 30 tasks and 16 nodes; (c) the result of running 150 generations of 60 tasks and 30 nodes; (d) the result of running 150 generations of 120 tasks and 50 nodes.

Figures 5(a)–5(d), respectively, show the network revenue comparison of task allocation when the sensor node is 8, and the task is 15, 25, 40, and 50. Use a histogram to see the gap more intuitively. The results of the four algorithms in Figures 5(a)–5(d) are the results of 150 generations. It can be seen that the total time of IACGA's revenue is greater than EGA, SA, and SFLA in the four cases, and the performance is always the best. When the number of sensor nodes is constant, as the number of tasks increases, the network revenue becomes larger. In Figure 5(d), it can be clearly seen that when the number of tasks is 50, the benefits of IACGA are far greater than EGA, SA, and SFLA. From Figures 5(a)–5(d), the same results can be obtained. IACGA optimizes ITWSN task allocation revenue performance better than EGA, SA, and SFLA.

Figure 6 shows the revenue growth percentages of 15 tasks and 8 nodes, 25 tasks and 8 nodes, 30 tasks and 16 nodes, 40 tasks and 8 nodes, 50 tasks and 8 nodes, 60 tasks and 30 nodes, and 120 tasks and 50 nodes, respectively. The data comes from Table 5. It can be seen from Figure 6 that under seven different tasks, the percentage of optimization of IACGA is greater than that of EGA, SA, and SFLA. In the case of 15 tasks, IACGA has the largest percentage increase in revenue, reaching 21%. When the number of tasks is 30 and 120, the improvement of IACGA is greater than that of EGA, SA, and SFLA. The seven comparison results show that IACGA has a better effect on task allocation network revenue improvement than EGA, SA, and SFLA.

Table 5 shows the network revenue values of IACGA, EGA, SA, and SFLA. It can be seen that as the number of tasks increases, the benefits of the four algorithms will also increase. Under different task allocation parameter settings, IACGA's network benefits are always the largest. The reason is that the adaptive mechanism and cloning mechanism we designed not only increase the global search capability but also speed up the search for the best individual. Compared with IACGA, EGA, SA, and SFLA tend to fall into the local optimum and have poor performance.

Table 6 shows the percentage of increase in IACGA, EGA, SA, and SFLA. When the node is 8, the maximum increase in revenue is 21%, and the task is 15. Followed by 16 nodes and 30 tasks. All the results in Table 5 show that under different conditions, the percentage increase of IACGA's revenue is always greater than that of EGA, SA, and

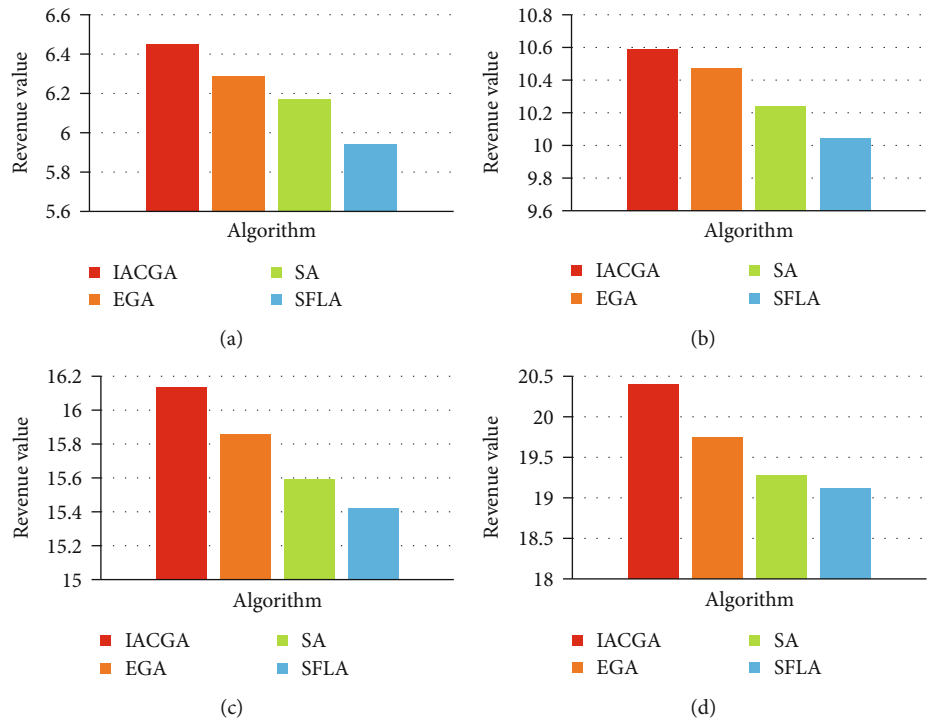

Figure 5: Comparison of the network revenue of different tasks of the four algorithms: (a) revenue of 8 nodes and 15 tasks; (b) revenue of 8 nodes and 25 tasks; (c) revenue of 8 nodes and 40 tasks; (d) revenue of 8 nodes and 50 tasks.

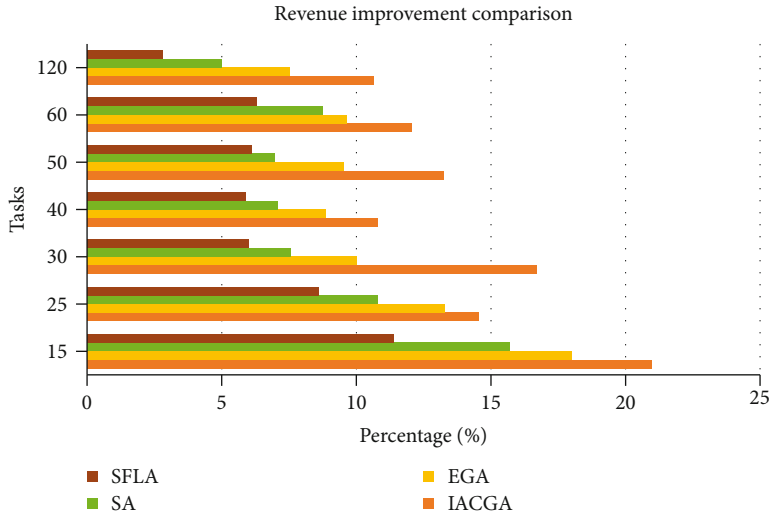

Figure 6: The comparison of revenue improvement of different numbers of tasks.

Table 5: The network revenue of the four algorithms.

Number of nodes and tasks	IACGA	EGA	SA	SFLA
15 tasks and 8 nodes	6.4493	6.2910	6.1700	5.9399
25 tasks and 8 nodes	10.5872	10.4715	10.2425	10.0398
40 tasks and 8 nodes	16.1344	15.8570	15.5931	15.4214
50 tasks and 8 nodes	20.4032	19.7414	19.2759	19.1211
30 tasks and 16 nodes	12.5777	11.8600	11.5961	11.4712
60 tasks and 30 nodes	22.5508	22.0611	21.8853	21.3917
120 tasks and 50 nodes	44.0117	42.7619	41.7644	40.8958

Table 6: The percentage increase of the four algorithms' revenue.

Number of nodes and tasks	IACGA	EGA	SA	SFLA
15 tasks and 8 nodes	21%	18%	16%	11%
25 tasks and 8 nodes	15%	13%	11%	9%
40 tasks and 8 nodes	11%	10%	7%	6%
50 tasks and 8 nodes	13%	9%	7%	6%
30 tasks and 16 nodes	17%	10%	8%	6%
60 tasks and 30 nodes	12%	10%	9%	6%
120 tasks and 50 nodes	11%	7%	5%	3%

Table 7: The running time of the four algorithms' 150 generations.

Number of nodes and tasks	IACGA	EGA	SA	SFLA
8 nodes and 15 tasks	0.10s	0.12 s	0.33 s	0.41 s
16 nodes and 30 tasks	0.15 s	0.15 s	0.47 s	0.58 s
30 nodes and 60 tasks	0.21 s	0.24 s	0.53 s	0.69 s
50 nodes and 120 tasks	0.34 s	0.41 s	0.75 s	0.94 s

Table 8: The computational complexity of the four algorithms is compared.

Algorithm	IACGA	EGA	SA	SFLA
Complexity	$O(n^2)$	$O(n^2)$	$O(n^2)$	$O(n^3)$

SFLA. The results show that IACGA is very effective in solving the task allocation problem.

Table 7 shows the running time of IACGA, EGA, SA, and SFLA under 4 different task allocation experimental parameters. The cycles of the four algorithms are all 150 generations. As the number of nodes and tasks increases, time increases. In the four cases, the running time of IACGA is less than that of EGA, SA, and SFLA. It is proved that the clone operator proposed in this paper accelerates the optimization time of the algorithm and finds the optimal solution quickly.

It can be seen from Table 8 that the algorithm complexity of IACGA, EGA, and SA is $O(n^2)$, and the algorithm complexity of SFLA is $O(n^3)$. IACGA and EGA have the same choice, crossover and mutation process; both are two recycle. In the process of SA temperature drop, each temperature level is also a double cycle. In the iterative process of SFLA, it is necessary to carry out grouping, loop within the group, loop outside the group, and individual coding loop. The complexity is a triple loop. Therefore, SFLA has the highest complexity among the four algorithms. Compared with EGA and SA algorithms, IACGA has the same order of complexity, but it can be seen from Figure 4 that the convergence speed and optimization performance of IACGA are better than those of EGA and SA.

The termination algebra of this algorithm is set to 150 generations. The convergence of the algorithm was discussed in the 150th generation. The convergence criterion of the algorithm adopts the range fluctuation percentage judgment method. The fluctuation range of this experiment is defined as 1%~2%. In the iterative process, we test that within 10 generations, the profit increase percentage ranges from 1% to 2%, which can be considered convergent. The four algorithms use the same convergence criteria. It can be seen from Figures 4(a)–4(d) that when the IACGA algorithm runs to the 60th generation, the upside of network revenue meets the convergence criterion. At this point, the network revenue has reached the highest value of the four algorithms. EGA, SA, and SFLA are also converging at this time, but the return value is very low. In 150 generations, these four algorithms converged to their respective approximate solutions.

In order to better verify the effectiveness of the proposed IACGA, this section compares the algorithm with EGA, SA, and SFLA under different numbers of tasks and experimental conditions of sensor nodes. The simulation data shows that the new IACGA proposed in this paper has great advantages in optimizing the network revenue of ITWSNs, and the algorithm runs fast and has strong optimization capabilities.

6. Conclusion

Aiming at the optimal task allocation scheme of intelligent transportation wireless sensor networks (ITWSNs), this paper proposes a new improved adaptive clone genetic algorithm (IACGA). Before the algorithm design, the task allocation model of the intelligent traffic sensor network is established. We design a new adaptive mechanism to control the probability of crossover and mutation to prevent the algorithm from falling into a local optimum. A new cloning mechanism is designed to select elite individuals to recombine a new population, which increases the diversity of the population and the optimization speed of the algorithm. In addition, we compare IACGA with EGA, SA, and SFLA by simulation and discussed in detail, which prove the superior performance of the proposed new algorithm, effectively solve the task allocation optimization problem in ITWSNs, and successfully maximize the ITWSN revenue.

Data Availability

The data presented in this study are available on request from the corresponding author. The data are not publicly available due to privacy.

Disclosure

The funders had no role in the design of the study; in the collection, analyses, or interpretation of data; in the writing of the manuscript; or in the decision to publish the results.

Conflicts of Interest

The authors declare no conflict of interest.

Acknowledgments

This paper was funded by the Corps innovative talents plan, grant number 2020CB001; project of Youth and middleaged Scientific and Technological In-novation Leading Talents Program of the Corps, grant number 2018CB006; the China Postdoctoral Science Foundation, grant number 220531; the

Funding Project for High Level Talents Research in Shihezi University, grant number RCZK2018C38; and the Project of Shihezi University, grant number ZZZC201915B.

References

[1] L. Zhu, F. R. Yu, Y. Wang, B. Ning, and T. Tang, "Big data analytics in intelligent transportation systems: a survey," *IEEE Transactions on Intelligent Transportation Systems*, vol. 20, no. 1, pp. 383–398, 2019.

[2] M. Veres and M. Moussa, "Deep learning for intelligent transportation systems: a survey of emerging trends," *IEEE Transactions on Intelligent Transportation Systems*, vol. 21, no. 8, pp. 3152–3168, 2020.

[3] D. Li, L. Deng, Z. Cai, B. Franks, and X. Yao, "Notice of retraction: intelligent transportation system in Macao based on deep self-coding learning," *IEEE Transactions on Industrial Informatics*, vol. 14, no. 7, pp. 3253–3260, 2018.

[4] Q. Wang, J. Zheng, H. Xu, B. Xu, and R. Chen, "Roadside magnetic sensor system for vehicle detection in urban environments," *IEEE Transactions on Intelligent Transportation Systems*, vol. 19, no. 5, pp. 1365–1374, 2018.

[5] G. Yan and Q. Qin, "The application of edge computing technology in the collaborative optimization of intelligent transportation system based on information physical fusion," *IEEE Access*, vol. 8, pp. 153264–153272, 2020.

[6] J. Wang, C. Jiang, Z. Han, Y. Ren, and L. Hanzo, "Internet of vehicles: sensing-aided transportation information collection and diffusion," *IEEE Transactions on Vehicular Technology*, vol. 67, no. 5, pp. 3813–3825, 2018.

[7] T. D. T. Nguyen, V. Nguyen, V. -N. Pham, L. N. T. Huynh, M. D. Hossain, and E. -N. Huh, "Modeling data redundancy and cost-aware task allocation in MEC-enabled internet-of-vehicles applications," *IEEE Internet of Things Journal*, vol. 8, no. 3, pp. 1687–1701, 2021.

[8] X. Hou, Z. Ren, J. Wang et al., "Reliable computation offloading for edge-computing-enabled software-defined IoV," *IEEE Internet of Things Journal*, vol. 7, no. 8, pp. 7097–7111, 2020.

[9] L. Sun, J. Wang, and B. Lin, "Task allocation strategy for MEC-enabled IIoTs via Bayesian network based evolutionary computation," *IEEE Transactions on Industrial Informatics*, vol. 17, no. 5, pp. 3441–3449, 2021.

[10] M. Okhovvat and M. R. Kangavari, "A mathematical task dispatching model in wireless sensor actor networks," *Computer Systems Science and Engineering*, vol. 34, no. 1, pp. 5–12, 2019.

[11] S. Khaliq, T. Maqsood, M. Ali, K. Bilal, S. A. Madani, and A. ur Rehman Khan, "A load balanced task scheduling heuristic for large-scale computing systems," *Computer Systems Science and Engineering*, vol. 34, no. 2, pp. 79–90, 2019.

[12] Y. Guo, F. Liu, N. Xiao, and Z. Chen, "Task-based resource allocation bid in edge computing micro datacenter," *Computers, Materials & Continua*, vol. 61, no. 2, pp. 777–792, 2019.

[13] D. Zhu, Y. Wang, C. You et al., "MMLUP: multi-source & multi-task learning for user profiles in social network," *Computers, Materials & Continua*, vol. 61, no. 3, pp. 1105–1115, 2019.

[14] X. Cao, H. Yu, and H. Sun, "Dynamic task assignment for multi-AUV cooperative hunting," *Intelligent Automation & Soft Computing*, vol. 25, no. 1, pp. 1–11, 2019.

[15] T. Ma, S. Pang, W. Zhang, and S. Hao, "Virtual machine based on genetic algorithm used in time and power oriented cloud computing task scheduling," *Intelligent Automation & Soft Computing*, vol. 25, no. 3, pp. 605–613, 2019.

[16] W. Lee, N. Vaughan, and D. Kim, "Task allocation into a foraging task with a series of subtasks in swarm robotic system," *IEEE Access*, vol. 8, pp. 107549–107561, 2020.

[17] C. Wei, Z. Ji, and B. Cai, "Particle swarm optimization for cooperative multi-robot task allocation: a multi-objective approach," *IEEE Robotics and Automation Letters*, vol. 5, no. 2, pp. 2530–2537, 2020.

[18] X. Tao and W. Song, "Location-dependent task allocation for mobile crowdsensing with clustering effect," *IEEE Internet of Things Journal*, vol. 6, no. 1, pp. 1029–1045, 2019.

[19] D. Yu, Y. Wang, and Z. Zhou, "Software crowdsourcing task allocation algorithm based on dynamic utility," *IEEE Access*, vol. 7, pp. 33094–33106, 2019.

[20] S. Liu and N. Wang, "Collaborative optimization scheduling of cloud service resources based on improved genetic algorithm," *IEEE Access*, vol. 8, pp. 150878–150890, 2020.

[21] K. Huang, Y. Dong, D. Wang, and S. Wang, "Application of improved simulated annealing genetic algorithm in task assignment of swarm of drones," in *International conference on information science, parallel and distributed systems (ISPDS)*, pp. 266–271, Xi'an, China, 2020.

[22] P. Y. Zhang and M. C. Zhou, "Dynamic cloud task scheduling based on a two-stage strategy," *IEEE Transactions on Automation Science and Engineering*, vol. 15, no. 2, pp. 772–783, 2018.

[23] J. Zhou, X. Zhao, X. Zhang, D. Zhao, and H. Li, "Task allocation for multi-agent systems based on distributed many-objective evolutionary algorithm and greedy algorithm," *IEEE Access*, vol. 8, pp. 19306–19318, 2020.

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

[25] U. F. Siddiqi, S. M. Sait, M. S. Demir, and M. Uysal, "Resource allocation for visible light communication systems using simulated annealing based on a problem-specific neighbor function," *IEEE Access*, vol. 7, pp. 64077–64091, 2019.