


Article

Task Planning of Space-Robot Clusters Based on Modified Differential Evolution Algorithm

Pengfei Xiao ¹, Hehua Ju ^{1,2,3,*}, Qidong Li ¹  and Feifei Chen ¹

¹ College of Astronautics, Nanjing University of Aeronautics and Astronautics, Nanjing 210000, China; xiaopengfei@nuaa.edu.cn (P.X.); liqidong@nuaa.edu.cn (Q.L.); fei_fei_ll@hotmail.com (F.C.)

² Key Laboratory Ministry of Industry and Information Technology, Nanjing 210000, China

³ Laboratory of Aerospace Entry, Descent and Landing Technology, Nanjing 210000, China

* Correspondence: juhehua@nuaa.edu.cn

Received: 7 May 2020; Accepted: 14 July 2020; Published: 21 July 2020



Abstract: This study studies the problem of on-orbit maintenance task planning for space-robot clusters. Aiming at the problem of low maintenance efficiency of space-robot cluster task-planning, this study proposes a cluster-task-planning method based on energy and path optimization. First, by introducing the penalty-function method, the task planning problem of the space-robot cluster under limited energy is analyzed, and the optimal-path model for task planning with comprehensive optimization of revenue and energy consumption are constructed; then, the maintenance task points are clustered to reduce the scale of the problem, thus reducing the difficulty of solving the problem; finally, a modified differential evolution algorithm is proposed to solve the problem of space-robot cluster task-planning, improve the performance of space-robot cluster task-assignment and path planning. Simulation results show that the proposed optimal-path model of space-robot cluster and the modified differential evolution algorithm can effectively solve the task-planning problem of spatial robot clusters.

Keywords: spatial robot clusters; task planning; task clustering; modified differential evolution algorithm

1. Introduction

With the continuous consumption of earth's energy, human vision has turned to space, and large-scale space solar-power stations (SSPS) have entered the vision of various space powers. On-orbit services, such as the assembly and maintenance of large space equipment [1,2] such as SSPS, are hot topics in current research. The mission planning of space robots plays an important role in on-orbit services. In recent years, research on space robot mission planning has made considerable progress [3,4], which has laid a solid foundation for the on-orbit assembly and maintenance of large space agencies.

Due to the limited efficiency of a single space robot, the collaborative operation of a cluster composed of multiple space robots is an effective way to complete large space tasks. Space-robot cluster task allocation and path planning are one of the core problems of multirobot system research [5–7]. Whether the task planning is reasonable is directly related to the final performance of the entire multirobot system. As the complexity and scale of the tasks performed increase, the effectiveness and efficiency of the task-planning method becomes increasingly important.

Aiming at the problem of collaborative task planning for robot clusters, many documents have proposed a series of models and algorithms [8–14], mainly including behavior-based allocation methods, market-mechanism methods, swarm-intelligence methods and idle chain-based methods. Zhen and Xing et al. [15] proposed an intelligent self-organizing algorithm to solve the problem of collaborative

search attack tasks of multiple agents. The architecture divides the global optimization problem into several local optimization problems. Each agent can solve its own local optimization problem, and then make the best decision for the multiagent system through the information exchange between the agents. Han and Li et al. [16] proposed a task planning algorithm based on the correlation between robot capabilities and team collaboration, under which an initialized associative array R was used to group the robots so that the robots in the group had better collaboration capabilities when performing tasks. Yao and Qi et al. [17] proposed an iterative strategy with an improved mission planning framework. By improving the onboard computing and communication resource management of agents, based on this strategy, each agent will reevaluate unreasonable planning results. Schwarzrock et al. [18] solved the task planning problem of multirobot systems by using group intelligence technology and proposed three complementary algorithms, forming a new method of increasing the amount of tasks executed. Gerkey and Mataric [19] developed a task-independent and system-independent classification method based on optimization theory, which is used to study task allocation problems. Moreover, its goal is to allocate limited resources in a way that effectively accomplishes certain tasks.

The problem of multirobot task assignment is a challenging research topic in the field of robotics [20–24]. The task assignment problem of the spacer-robot cluster can be transformed into an optimization problem, and then some intelligent-optimization methods can be used to solve the optimization problem. Hussein and Marinplaza et al. [25] proposed a general method to solve the problem of multirobot task assignment (MRTA). This method combines simulated annealing and genetic algorithm and can appropriately use all available resources to optimize the allocation, thereby improving the overall system performance and reducing costs. Zheng and Li [26] proposed an optimized multirobot task-allocation and scheduling method based on fish swarm algorithm. Chen and Zhang et al. [27] proposed a memetic algorithm based on ant colony optimization. At the same time, the problem of the maximum travel distance and total travel distance of multiple robots is optimized.

Due to the strong similarity between the MRTA problem and the multiple traveling salesman problem, some researchers regard the MRTA problem as an example of the multiple traveling salesman problem. The traveling salesman problem is one of the standard combined discrete optimization problems. There are two methods for solving the TSP: Exact method and heuristic method. Khan and Maiti [28] adopted multiple update rules and K-opt operations to improve the artificial bee colony algorithm, they used the characteristics of the exchange sequence to exchange the urban sequence (solution/path) of the problem, and created update rules of different solutions (path) for the algorithm to solve the travel agent problem. Trigui and Cheikhrouhou et al. [29] proposed a centralized solution based on fuzzy logic algebra and combined with two goals: taking the minimum total travel distance of the traveling salesman and the maximum travel distance of any robot as the goal. Mahi et al. [30] combined the characteristics of particle swarm optimization, ant colony algorithm and 3-opt algorithm, and proposed a hybrid algorithm PSO-ACO-3-opt for solving standard TSPs.

The task-planning method of the spacer-robot cluster will directly affect the efficiency of the entire system. Therefore, choosing the optimal robot cluster task-planning method becomes the key to effectively solve the task planning problem of the robot cluster system. The existing methods usually consider the task planning problem of robot clusters as a multiple traveling salesman problem and establish the shortest-path model of robot cluster task-planning. When using the shortest-path model to solve space maintenance tasks, there exist the following problems: (1) no modeling analysis of space-robot cluster task-planning problems under resource constraints; (2) no consideration of the maximum benefit of space maintenance when space resources are limited; (3) no considering the difficulty of solving large-scale planning problems.

Based on the analysis of the current research situation, this study proposes a new space-robot cluster task-planning scheme for the maintenance of SSPS. First, the task planning of the space-robot cluster under limited energy is analyzed, and a task planning model of space-robot clusters is proposed with penalty function added, which is comprehensively optimal for the consideration of income and energy consumption. Then, the task-clustering method is used to cluster the maintenance tasks to

reduce the problem size and reduce the problem solving time; finally, a modified differential evolution algorithm including multiple neighborhood operations, roulette, de-crossover, multiple groups and other strategies is proposed to solve the problem of task allocation and path planning of multi-robots, complete the mission planning of the space-robot cluster, and improve the maintenance efficiency.

The rest of the paper is organized as follows: Section 2 introduces the background of basic space maintenance tasks and the theoretical basis of modeling. Section 3 proposes the modeling theory of space-robot cluster task-planning. In Section 4, a modified differential evolution algorithm is proposed, and the algorithm is described in detail. In Section 5, the accuracy and effectiveness of the proposed model and algorithm are verified by experiments, and the results are discussed. Section 6 presents the conclusion.

2. Theory Background

2.1. Space Solar Power Station Model

SSPS [31,32] is based on space solar-power generation technology, which can provide constant and pollution-free energy. The SSPS mainly consists of three parts: solar power generation device, energy conversion and transmission device, ground receiving and conversion device. The space above the atmosphere is not affected by the weather, nor is it absorbed or scattered by the atmosphere. The solar energy received is far more than the ground, and the solar-power station can track the sun at any time on the orbit, and keep it in working condition.

The array of solar cells that convert energy into electricity at a SSPS consists of thousands of solar cells. Solar cells are made of pure silicon element blocks. Due to the long service cycle of the SSPS, it is easy to cause damage to parts under the condition of high radiation and large temperature variation in space. Aiming at the maintenance problem of large SSPS, this paper proposes the task planning scheme of space-robot cluster.

2.2. Basic Differential Evolution Algorithm

The differential evolution algorithm (DE) is an efficient evolutionary optimization technology, first proposed by Storn and Price [33]. DE is a direct-search method based on population. In this section, we will briefly explain the working process of DE.

Step 1: Initialization

DE algorithm has a population composed of individual vectors with NP dimensions of D , where each individual is represented as: $x_i^g = \{x_{i,1}^g, x_{i,2}^g, \dots, x_{i,j}^g, \dots, x_{i,D}^g\}$, $i = 1, 2, \dots, NP$, g expressed evolutionary algebra. Set the lower limit of the parameter variable is $X_{min} = \{x_{min,1}, \dots, x_{min,D}\}$ and the upper limit is $X_{max} = \{x_{max,1}, \dots, x_{max,D}\}$.

$$x_{i,j}^0 = rand(0, 1) \cdot (x_{max,j} - x_{min,j}) + x_{min,j}, \quad j = 1, 2, \dots, D \tag{1}$$

Among them, $rand(0, 1)$ represents the generation of uniform random Numbers between $[0, 1]$.

Step 2: Variation

According to several different mutation strategies commonly used in the literature, the following operations are performed for each target vector x_i^g ($i = 1, 2, \dots, NP$):

1. DE/rand/1

$$v_i^{g+1} = x_{r1}^g + F \cdot (x_{r2}^g - x_{r3}^g) \tag{2}$$

2. DE/rand/2

$$v_i^{g+1} = x_{r5}^g + F \cdot (x_{r1}^g - x_{r2}^g) + F \cdot (x_{r3}^g - x_{r4}^g) \tag{3}$$

3. DE/best/1

$$v_i^{g+1} = x_{best}^g + F \cdot (x_{r1}^g - x_{r2}^g) \tag{4}$$

4. DE/best/2

$$v_i^{g+1} = x_{best}^g + F \cdot (x_{r1}^g - x_{r2}^g) + F \cdot (x_{r3}^g - x_{r4}^g) \tag{5}$$

5. DE/current-to-best/1

$$v_i^{g+1} = x_i^g + F \cdot (x_{best}^g - x_i^g) + F \cdot (x_{r1}^g - x_{r2}^g) \tag{6}$$

In the formula: v_i^{g+1} represents the i 'th individual in the population obtained after mutation, the randomly selected sequence $r1, r2, r3, r4, r5$ and the target vector sequence i are different from each other, x_{best}^g is the best performing individual in the g generation. F is the mutation operator, which controls the amplification of the deviation vector.

Step 3: Cross

In order to increase the diversity of interference vectors, crossover operation is introduced, and the test vector is:

$$u_i^{g+1} = (u_{i,1}^g, u_{i,2}^g, \dots, u_{i,j}^g, \dots, u_{i,D}^g) \tag{7}$$

$$u_{i,j}^{g+1} = \begin{cases} u_{i,j}^{g+1} & \text{if } (rand(0, 1) \leq CR) \text{ or } j = j_{rand} \\ x_{i,j}^{g+1} & \text{otherwise} \end{cases} \tag{8}$$

Among them, $rand(0, 1)$ is the random number between $[0, 1]$, CR is the crossover factor, which is a constant within the range of $[0, 1]$; the larger the value of CR , the greater the possibility of crossover; j_{rand} is an integer randomly selected within the range of $[1, D]$, which can ensure that the experimental individual u_i^{g+1} obtains at least one element from the mutant individual v_i^{g+1} .

Step 4: Select

By calculating the fitness of the experimental vector, and then through the selection mechanism of greedy thought, the original population and the experimental population are compared one-to-one, and the next generation is selected to ensure that the next generation is better than the previous generation.

2.3. Clustering Algorithm

Due to the large scale of space solar-power stations, there are many nodes to be repaired, which greatly increases the difficulty of space-robot cluster to perform maintenance tasks. The current task planning algorithm is not very effective for planning large-scale problems, and as the number of tasks increases, the planning time increases significantly. Therefore, how to reduce the scale of the problem is one of the elements to solve the task planning problem. This study uses clustering algorithm to reduce the scale of the problem and reduce the difficulty of solving the planning problem.

Clustering has been extensively studied for many years, and researchers have proposed many clustering algorithms [34–37]. For large-scale space problems, space task segmentation can be effectively divided into disjoint subsets, and only allow each space robot to select from subsets of the whole task set. According to the target of space maintenance task, a group of data points is divided into several clusters by cluster analysis, and then the task planning of space-robot cluster is carried out for the maintenance points after clustering, so as to reasonably arrange and organize space robots.

The movement of the space robot is divided into the movement of the space robot base and the movement of the mechanical arm. For space maintenance operations, the movement of the satellite base moves between the clustering centers points, and the space robot only needs to operate the mechanical arm to complete the maintenance task within the clustering scope, without moving the base of the space robot. According to the results after clustering, the task planning of space robots is

carried out. This hierarchical planning idea can effectively reduce the planning complexity and shorten the planning time.

If point P_x is taken as the center of the clustering, when the distance between point P_x and point P_i is less than eps , then point P_i is put into set $Cluster_x^1$, as shown in Equation (9), and the rest points are followed in turn.

$$Cluster_x^k = \begin{cases} P_i | Dist(i, x) < eps & k = 1 \\ P_j | P_j \in Cluster_x^{k-1}, Dist(i, j) < eps & k > 1 \end{cases} \quad (9)$$

$$Cluster_x^{area} = Cluster_x^1 \cup Cluster_x^2 \cup \dots \cup Cluster_x^k$$

Among them: eps represents the clustering radius, and k represents the number of layers of clustering, which can be set according to the requirements.

Repeat the above process to obtain other clustering sets. The set of regional maintenance targets clustered by point targets is $Task = \{Cluster_x^{area}, Cluster_y^{area}, \dots, Cluster_M^{area}\}$, and M is the total number of clusters.

3. Task Planning Model of Space-Robot Cluster

Energy in space is a scarce resource. For space robots, it is necessary to consider that the task completed is optimal when the energy is constant. In this paper, the optimal-path model of a space-robot cluster under resource constraints is proposed, which can easily solve the task planning model of spatial robot cluster. In this paper, each space robot is regarded as an independent task executor, which has the basic characteristics of an intelligent agent. The space-robot cluster is expressed as $Robot = \{R_1, R_2, \dots, R_i, \dots, R_N\}$, $1 \leq i \leq N$, in which N represents the total number of space robots. The space maintenance task cluster is expressed as $Task = \{T_1, T_2, \dots, T_j, \dots, T_M\}$, $1 \leq j \leq M$, where M represents the total number of spatial clustering task sets; $T_j = \{t_1, t_2, \dots, t_k, \dots, t_{num}\}$, $1 \leq k \leq num$, where num represents the total number of tasks in each cluster.

3.1. Constraints

The problem of collaborative task allocation of space-robot clusters aims at designing charging piles for space robots at specific locations in accordance with the principle of sustainable work during the execution of tasks. Therefore, the path of space robot should be a closed path starting from the charging pile position. The space robot needs to satisfy some constraints to complete the maintenance task: reserve energy, number of space robots, closed path and so on.

Defining variables

$$x_{ij}^k = \begin{cases} 1 & \text{If robot } k \text{ goes through node } i \text{ to node } j \\ 0 & \text{else} \end{cases}, y_i^k = \begin{cases} 1 & \text{If robot } k \text{ repairs node } i \\ 0 & \text{else.} \end{cases}$$

The energy consumed by the k' th robot performing the task can be expressed as

$$Consumption_k = \sum_{i=0}^m (\alpha_k \cdot dis_{i,i+1}^k + \beta_k \cdot num_{i+1}^k) + \alpha_k \cdot dis_{m,0}^k \quad (10)$$

where in:

$$dis_{i,i+1}^k = c_{ij} \cdot x_{ij}^k \quad (11)$$

$$\sum_{k=1}^N y_i^k \leq 1, i = 1, 2, \dots, M \quad (12)$$

$$\sum_{i=0}^M x_{ij}^k = y_j^k \quad \forall k \in \{1, 2, \dots, N\}, \forall j \in \{1, 2, \dots, M\} \quad (13)$$

$$\sum_{j=0}^M x_{ij}^k = y_i^k \quad \forall k \in \{1, 2, \dots, N\}, \forall i \in \{1, 2, \dots, M\} \tag{14}$$

where m represents the number of cluster nodes on the space robot path, α_k represents the energy consumption per unit distance of the k 'th robot, $dis_{i,i+1}^k$ represents the path length of the k 'th space robot from the i 'th node to the $(i + 1)$ 'th node on the path, c_{ij} represents the distance between node i and node j , $i = 0$ represents the starting position of the robot, β_k is the energy consumption of the k 'th robot repairing a single maintenance node, num_{i+1}^k represents the number of maintenance nodes in the $(i + 1)$ 'th cluster node on the path of the k 'th space robot. Constraint (12) ensures that each maintenance point has at most one space robot for maintenance; constraint (13) ensures that each node in the maintenance path has only one starting point; constraint (14) ensures that the node in each maintenance path has only one end point. In general, these constraints can ensure that each maintenance point is assigned to at most one space robot, and the space robot passes all the assigned nodes and walks a closed loop.

In order to ensure that the robot can safely return to the starting point, the energy of the space robot is greater than that of the charging pile at the last maintenance point. Each robot should return to the charging pile position before the safe energy of the battery is exhausted, namely:

$$Consumption_k < \xi \cdot Energy_k \tag{15}$$

In which $Energy_k$ is the total energy of the k 'th robot, ξ is the safety factor.

3.2. Objective Function

The task planning of space-robot cluster mainly considers the benefits and costs of completing the tasks of each node with limited energy. For simplicity, we only consider the moving distance of space robots and the cost of repairing solar panels. Assuming that the damage degree of solar panel is divided into three levels, with damage level I for 10–40%, II for 40–70% and III for 70–100% and the maintenance benefit increases with the increase of damage degree.

The goal of space-robot cluster task-planning is to maximize the overall benefit

$$J = \max(\omega_1 \cdot \sum_{k=1}^N \sum_{i=1}^m benefit_i^k + \omega_2 \cdot \sum_{k=1}^N Consumption_k) \tag{16}$$

subject to

$$benefit_i^k = gain \cdot \sum_{j=1}^{num_i^k} damage_j \tag{17}$$

$$Consumption_k < \xi \cdot Energy_k \quad \forall k \in \{1, 2, \dots, N\} \tag{18}$$

$$damage_j \in \{1, 2, 3\} \tag{19}$$

where ω_1 and ω_2 respectively represent the weight coefficients of maintenance revenue and maintenance cost, $benefit_i^k$ is the maintenance revenue in the i 'th cluster node of the k 'th robot, $damage_j$ is the damage level and $gain$ is the profit coefficient.

In this model, the objective function (16) aims to maximize the value of the profit and the maximum revenue and the minimum energy consumption of completing the task must be considered. Moreover, the energy consumption includes the maintenance path consumption and the maintenance node consumption; constraint (18) represents the energy limit of each space robot.

According to the research on constraint-processing methods, constraint-processing methods are mainly divided into three categories: penalty-function methods, feasible-rule methods and multi-objective methods. The penalty-function method is to add a penalty function to the objective

function. Because of its simple principle and easy implementation, it is one of the most widely used constraint-processing methods. In this paper, by analyzing the task planning model of space robot, the concept of penalty function is introduced to transform the constraint problem into an unconstrained problem.

By introducing the penalty function, the objective function is

$$\max J = \omega_1 \cdot Profit + \omega_2 \cdot Cost + Penalty \tag{20}$$

In which

$$\begin{aligned} Profit &= \sum_{k=1}^N \sum_{i=1}^m benefit_i^k \\ Cost &= \sum_{k=1}^N Consumption_k \\ Penalty &= \varphi \cdot \sum_{k=1}^N (Consumption_k - \xi \cdot Energy_k) \end{aligned} \tag{21}$$

In which: ω_1, ω_2 respectively represent the maintenance revenue and the maintenance cost, φ represents the penalty coefficient.

4. Modified Differential Evolution Algorithm

Since the emergence of differential evolution algorithm, scholars have done much research work and achieved many results [38–42]. DE algorithm is an evolutionary algorithm based on population optimization, which mainly searches by differential variation, crossover and selection operations and can be used to solve global optimization problems. This paper proposes some improvements for the difference algorithm to solve the problem of space-robot cluster collaborative task planning.

4.1. Adaptive Control Parameter Strategy

In the search process of the basic differential evolution algorithm, the mutation operator takes a constant, but in practice, the value of the mutation operator is difficult to determine, if the value of the mutation operator is too large, the search efficiency of the algorithm is reduced, and the accuracy of the global optimal solution is low; if the mutation operator is too small, the diversity of the population is prone to premature convergence. This paper uses a strategy with an adaptive mutation operator.

$$F = F_0 \cdot 2^\kappa, \quad \kappa = e^{1 - \frac{gMax}{gMax+1-g}} \tag{22}$$

Among them, F_0 is the initial mutation operator, $gMax$ is the maximum evolutionary algebra, g is the current evolutionary algebra. In the early stage of the algorithm, the adaptive mutation operator has a large value, which keeps the diversity of individuals. In the later stage of the algorithm, the mutation operator decreases gradually, retains good information and increases the probability of searching the global optimal solution.

4.2. Roulette Selection Strategy

In order to increase the probability of excellent individuals being selected, roulette selection is used to maintain the principle that the higher the fitness is, the higher the probability of being selected. The roulette method completely relies on random numbers for selection each time, which increases the uncertainty of selection. The basic idea of roulette selection is that the probability of each individual being selected is proportional to its fitness, so the probability of each individual being selected can be obtained by calculating the fitness ratio.

$$p(x_i^g) = \frac{f(x_i^g)}{\sum_{j=1}^{NP} f(x_j^g)}, \quad i \in \{1, 2, \dots, NP\} \tag{23}$$

Among them, $p(x_i^g)$ is the probability that individual x_i^g is selected in the next generation evolution and $f(x_i^g)$ is the fitness function value corresponding to individual x_i^g .

Then calculate the cumulative probability of each individual, that is, the sum of the selection probabilities of all individuals.

$$q_i = \sum_{j=1}^i p(x_j^g) \tag{24}$$

Moreover, then randomly generate $r \in [0, 1]$, if $q_i > r$, then select individuals x_i^g .

In this study, the roulette operation is performed on the population after mutation and crossover and the optimal individual generated by the roulette selection is used to participate in the next generation of group evolution, which can improve the convergence efficiency of the algorithm.

4.3. Multi-Neighbor Operation Strategy

In the early stage of the DE algorithm iteration, the individual differences in the population are large, and the mutation operation will make the algorithm have a strong global search ability; by the middle of the iteration, its optimal values tend to be consistent. In order to improve the diversity of DE algorithm individuals and avoid the premature convergence of DE algorithm and the problem of local extremes, this study proposes a multi-neighbor operation strategy. For individuals in the population, the multiple pairs of positions are exchanged randomly and the value at each position of the exchange is the node on the exchange path.

$$x_i^g = \begin{cases} \text{exchange}(x_{i,p1}^g, x_{i,p2}^g) & \text{if } (\text{rand} \leq \gamma) \\ \text{exchange}(x_{i,p1}^g, x_{i,p2}^g, \dots, x_{i,pn}^g) & \text{otherwise} \end{cases} \tag{25}$$

where, γ is the neighborhood factor, $n \in [3, D/2]$.

4.4. De-Crossover Strategy

In the planning process, in order to increase the convergence speed, it is necessary to design a de-crossover strategy to reduce cross paths. First of all, it is necessary to judge whether the two line segments ab and cd cross, $a(x_1, y_1)$, $b(x_2, y_2)$, $c(x_3, y_3)$, $d(x_4, y_4)$ and the two lines are:

$$ab : \begin{cases} x = x_1 + m \cdot (x_2 - x_1) \\ y = y_1 + m \cdot (y_2 - y_1) \end{cases} \quad 0 \leq m \leq 1 \tag{26}$$

$$cd : \begin{cases} x = x_3 + n \cdot (x_4 - x_3) \\ y = y_3 + n \cdot (y_4 - y_3) \end{cases} \quad 0 \leq n \leq 1 \tag{27}$$

Set, $D = \begin{vmatrix} x_2 - x_1 & x_3 - x_4 \\ y_2 - y_1 & y_3 - y_4 \end{vmatrix}$ when $D = 0$, two line segments ab and cd are parallel or coincide; when, $D \neq 0$ the calculation formula of m and n are:

$$m = \frac{1}{D} \begin{vmatrix} x_3 - x_1 & x_3 - x_4 \\ y_3 - y_1 & y_3 - y_4 \end{vmatrix} \tag{28}$$

$$n = \frac{1}{D} \begin{vmatrix} x_2 - x_1 & x_3 - x_1 \\ y_2 - y_1 & y_3 - y_1 \end{vmatrix} \tag{29}$$

When $0 \leq m \leq 1$ and $0 \leq n \leq 1$, the paths ab and cd have an intersection, otherwise none.

Five nodes are used to illustrate the crossover strategy, as shown in Figure 1. The specific process is as follows, starting from node A , walk a closed loop along $ABCDEA$, where AB and DE cross,

then replace with AD and BE and chain node B to node D in reverse order, so the loop $ADCBEA$ is formed. The specific operation is shown in the pseudocode Algorithm 1.

Algorithm 1. The pseudo-code of uncrossed strategy.

Input: path (A, B, C, D, E, A)
Output: path_uncross (A, D, C, B, E, A)

1. **for** $i \leftarrow 1$:length(path)
2. $j \leftarrow i + 1$
3. **While** $j < \text{length}(\text{path})$
4. $ab \leftarrow \text{path}(i, i+1)$
5. $cd \leftarrow \text{path}(j, j+1)$
6. $\text{flag} \leftarrow \text{Cross_judgement}(ab, cd)$ % Judge whether cross
7. **if** $\text{flag} = 1$
8. $\text{path}(i+1:j) \leftarrow \text{path}(j:-1:i+1)$
9. **end if**
10. $j \leftarrow j+1$
11. **end While**
12. **end for**
13. $\text{path_uncross} \leftarrow \text{path}$
14. **return** path_uncross

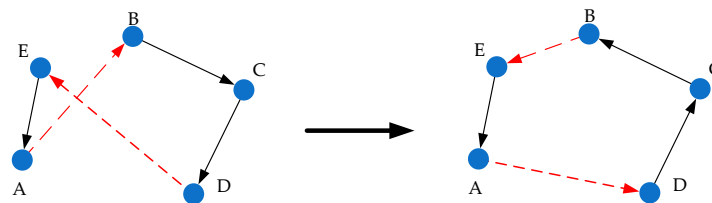


Figure 1. Schematic diagram of uncrossed strategy.

4.5. Multi-Population Integration Strategy

In order to solve the problem that differential evolution algorithm is prone to local optimal solution and premature maturity, this study proposes a multi-population differential evolution algorithm. The algorithm splits the initial large population into multiple subpopulations, and each subpopulation is assigned to a mutation strategy, so as to realize parallel exploration among subpopulations, thereby enhancing the global exploration ability of the algorithm. With iteration, the reward subpopulations are assigned to the best performing mutation strategies in previous generations in order to obtain more computing resources and gradually enhance the algorithm's ability to locally explore excellent solutions.

For population 1, population 2 and population 3, we adopted different strategies, our principle is to select well-researched mutation strategies, and each strategy has its own advantages. The three mutation strategies selected in this paper include "DE/Rand-to-pbest/1", "DE/current-to-RAND/1" and "DE/pbad-to-pbest/1". Among them, "DE/current-to-pbest/1" uses multiple excellent individual information to make the algorithm not converge prematurely, and the base vector is randomly selected, so its global search ability is relatively strong; "DE/current-to-rand/1" is an arithmetic crossover, which has the characteristics of constant rotation and has advantages in solving the rotation problem; "DE/pbad-to-pbest/1" can make use of good solution information and it can utilize the information of the bad solution (pbad) toward the good solution to balance exploration and exploitation. In this paper, the global and local exploration ability of the algorithm is effectively guaranteed by this mechanism. The pseudocode for the multi-population integration strategy is shown in Algorithm 2.

Algorithm 2. Pseudo-code of multi-population integration strategy.

```

1. Set up the  $\Delta f_j \leftarrow 0$  each  $j \in \{1, 2, 3, 4\}$ 
2. Initialize the  $NP, ng$  (Change the reward population per  $ng$  generation)
3. Randomly initialize  $pop$  to randomly distribute it in the search space
4. Initialize the  $\lambda_j$  and set up the  $NP_j \leftarrow \lambda_j \cdot NP$ 
5. Set up the  $g \leftarrow 0$  and  $gMax$ 
6. while  $g < gMax$  do
7.      $g \leftarrow g + 1$ ;
8.     if  $\text{mod}(g, ng) = 0$ 
9.          $k \leftarrow \arg(\max_{1 \leq j \leq 3} (\Delta f_j / (ng \cdot NP_j)))$ 
10.         $\Delta f_j \leftarrow 0$ 
11.    end if
12.    According to  $NP_j$ ,  $pop$  is randomly divided into  $pop_1, pop_2, pop_3, pop_4$ 
13.    Set up the  $pop_k \leftarrow pop_k \cup pop_4$  and  $NP_k = NP_k + NP_4$ 
14.    for  $j \leftarrow 1$  to 3
15.        for  $i \leftarrow 1$  to  $NP_j$ 
16.            switch  $j$ 
17.                case 1: %Perform the first mutation strategy DE/rand-to-pbest/1 in  $pop_1$ 
18.                     $v_i^{g+1} \leftarrow x_i^g + K \cdot (x_{best}^g - x_i^g) + F \cdot (x_{r1}^g - x_{r2}^g)$ ,  $i \in \{1, 2, \dots, NP\}$ 
19.                case 2: %Perform the second mutation strategy DE/current-to-rand/1 in  $pop_2$ 
20.                     $v_i^{g+1} \leftarrow x_i^g + F \cdot (x_{r1}^g - x_i^g) + F \cdot (x_{r2}^g - x_{r3}^g)$ ,  $i \in \{1, 2, \dots, NP\}$ 
21.                case 3: %Perform the third mutation strategy DE/pbad-to-pbest/1 in  $pop_3$ 
22.                     $v_i^{g+1} \leftarrow x_i^g + F \cdot (x_{best}^g - x_{pbad}^g)$ ,  $i \in \{1, 2, \dots, NP\}$ 
23.            end switch
24.            Perform crossover operations and boundary condition processing
25.            if  $f(X_{i,g}) < f(U_{i,g})$ 
26.                 $X_{i,g+1} \leftarrow X_{i,g}$ 
27.            else
28.                 $X_{i,g+1} \leftarrow U_{i,g}$ ,  $\Delta f_j \leftarrow \Delta f_j + f(X_{i,g}) - f(U_{i,g})$ 
29.            end if
30.        end for
31.     $pop \leftarrow pop_1 \cup pop_2 \cup pop_3$ 
32. end while

```

4.6. Algorithm Flow

For task planning of space-robot cluster, the overall planning process is shown in Figure 2.

1. First, generating the coordinates of the maintenance node, the maintenance level and the revenue and determine the number of space robots and the origin coordinates;
2. Cluster the maintenance nodes to obtain the coordinates of the cluster center point, the revenue within the cluster and the number of target points within the cluster;
3. Generate the energy loss matrix between clusters and the distance energy loss matrix from the space robot to each cluster center;
4. Modified differential evolution algorithm.
 - (1) Initialize the multi-population parameters;
 - (2) Randomized population;
 - (3) Adaptive differential evolution operator $F = F_0 \cdot 2^\kappa$, $\kappa = e^{1 - \frac{gMax}{gMax+1-g}}$;
 - (4) The reward subpopulation is assigned to the k 'th population, set $pop_k = pop_k + pop_4$ and $NP_k = NP_k + NP_4$, where $k = \{1, 2, 3\}$;
 - (5) pop_1, pop_2, pop_3 carry out their respective strategy differential evolution operations;

- (6) Population combination, judge whether the end condition is satisfied and output the robot cluster optimization result if it is satisfied, otherwise repeat steps (3)–(5).

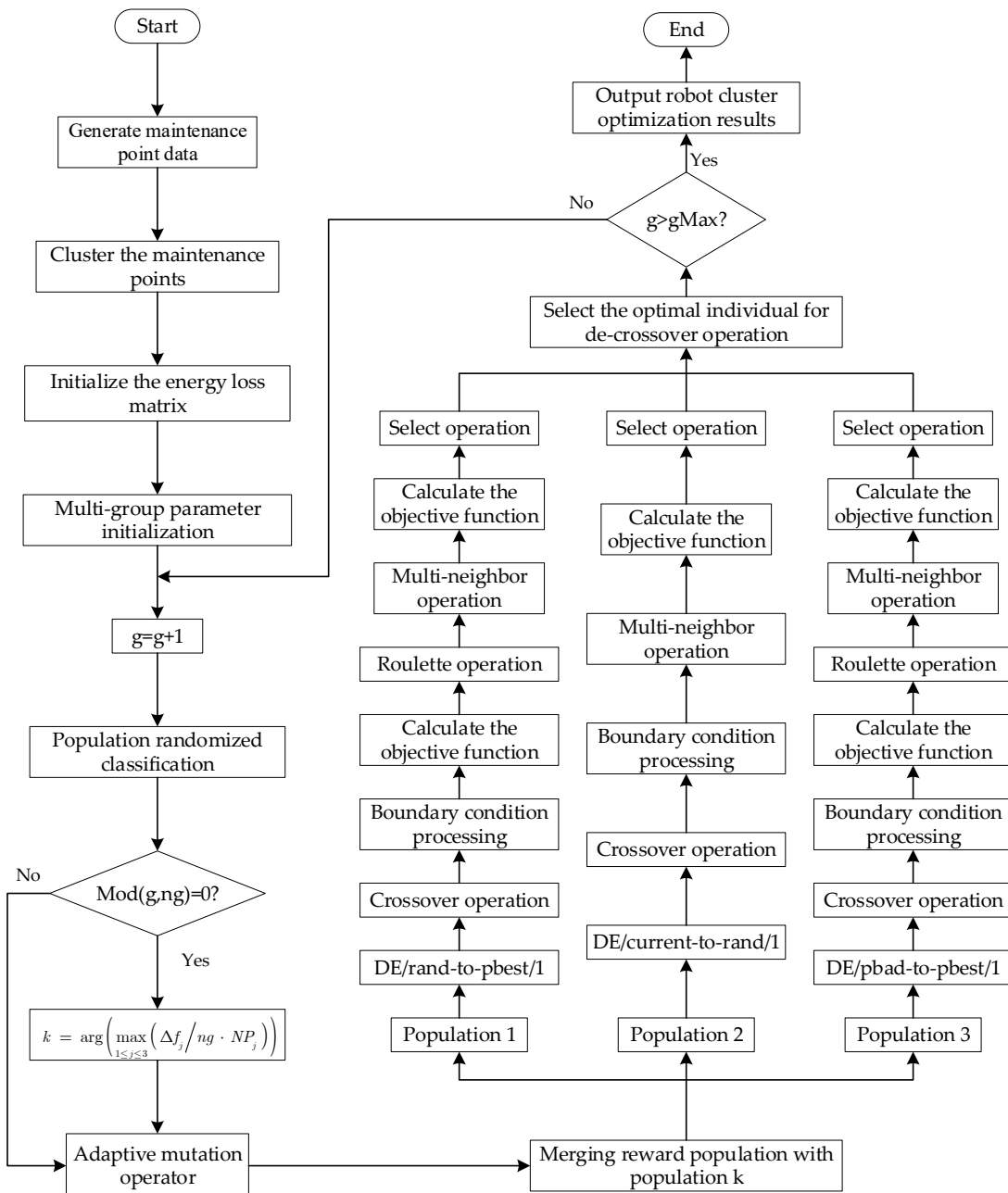


Figure 2. Flow chart of the algorithm.

5. Experimental Results

The effectiveness of the algorithm is verified by computer simulation. The simulation was implemented in Matlab 2016a, and the computer parameters were Intel Core i7-8750 processor, 2.20 GHz CPU and 8 GB RAM memory configuration. The basic parameters of the simulated SSPS are as follows: the size of the maintenance area is 180 m × 60 m, the size of the solar panel is 0.5 m × 0.6 m and the damage rate of the solar panel is between 2%–2.5%.

5.1. Space Maintenance Point Clustering

Due to the different degrees of damage to the solar panels in the space solar-power station, their maintenance priorities are also different. For the degree of each solar panel damage Class I, Class II and Class III, the benefits increase as the degree of damage increases. This paper takes the maintenance of solar array as an example, and performs task clustering on 800 maintenance point targets, so as to test the performance of the algorithm. The target coordinates of maintenance points are input into the clustering algorithm, and the maintenance points are clustered into regional target clusters and single maintenance point clusters by the clustering algorithm. The results of the clustering algorithm are shown in Table 1, (x, y) represents the position coordinates of each cluster center, Number of points in the cluster (NPC) is the number of points that need to be repaired in each cluster area, and the Income within cluster (IC) represents the sum of the income of all maintenance points within the cluster. As shown in Equation (17), this paper sets the profit coefficient $gain$ to 10. According to Equation (9), we set $eps = 0.5$ (half of the robotic arm length), $k = 2$.

When the maintenance point in the cluster is greater than or equal to 3, it is the regional target clustering—and when it is less than 3—it is the noise point in the clustering algorithm, namely, the single target clustering. As can be seen from Table 1, the clustering results include 92 regional target clustering and 8 single target clustering, and the total income of 800 maintenance points is 15,870. The clustering algorithm is used to reduce 800 maintenance nodes to 100 clustering centers, which effectively reduces the scale of planning problems and reduces the difficulty of solving the planning problems.

5.2. Comparison Experiment of Optimal-Path Model Optimization

5.2.1. Experimental Comparison

In the process of task planning, it is necessary to reasonably consider the relationship between the paths between clusters and the number of points within the clusters—and then to complete the optimization of the robot clusters revenue and energy consumption under a certain energy. In this experiment, under the condition of certain energy, the newly proposed optimal-path model of space-robot cluster and the shortest-path model of space-robot cluster are compared. The optimal path is solved by the modified differential evolution algorithm, and the total number, revenue and completion rate of the maintenance tasks under the condition of certain energy are compared.

In order to simplify problem solving, it is assumed that each space robot's movement consumption α_k and maintenance failure point consumption β_k are the same, set $\alpha_k = 5$, $\beta_k = 1$, $k = 1, 2, 3, 4, 5$. In this experiment, the corresponding parameters are set as: $NP = 500$, $ng = 20$, $\gamma = 0.9$, $CR = 0.1$, $F_0 = 0.4$, $\lambda_1 = \lambda_2 = \lambda_3 = \lambda_4 = 0.25$, $gMax = 10,000$. The clustering results of 800 maintenance points are taken as the target node to be planned in this experiment. The charging piles in each space are the starting point of each space robot. The space robots are required to return to the starting position after repairing each target node according to the design path. The initial positions of the five space robots are, respectively $(0, 0)$, $(-90, 30)$, $(-90, -30)$, $(90, 30)$, $(90, -30)$.

In this experiment, the total battery capacity of the space-robot cluster is set to 5000, 6000 and 7000, respectively. The energy of each robot is shown in Table 2. Considering the optimal-path model and the shortest-path model of the robot cluster, the maximum number, the maximum revenue and the task completion rate of the space-robot cluster maintenance solar array are calculated. The maintenance path and completed tasks are shown in Tables 3–5, respectively. In this experiment, the safety factor, maintenance revenue weight, maintenance cost weight and penalty function coefficients are set as $\xi = 0.8$, $\omega_1 = 1$, $\omega_2 = -0.5$, $\varphi = -400$. Consumption is the energy consumption of each space robot. Income is the total income of repair points on the repair path. Number of tasks is the number of tasks completed by the space-robot cluster at a given energy. Income rate is the ratio of the income completed by the space-robot cluster and the total task income.

Table 1. Information after clustering of space tasks.

Num	(x, y)	NPC	IC	Num	(x, y)	NPC	IC
1	(−88.5, −28.2)	6	110	51	(51.5, 24.0)	9	170
2	(−23.5, −12.0)	18	350	52	(40.0, 2.4)	11	220
3	(−80.0, −6.6)	13	220	53	(68.5, 9.6)	3	80
4	(−46.5, −6.0)	8	130	54	(56.5, 4.2)	16	340
5	(−56.5, −8.4)	7	130	55	(−82.0, −24.0)	4	70
6	(−40.0, −16.8)	14	310	56	(−75.5, 2.4)	9	220
7	(−13.5, −13.2)	7	110	57	(−63.0, −2.4)	5	80
8	(−10.0, −3.6)	15	300	58	(2.5, −8.4)	5	120
9	(−6.5, −18.6)	9	160	59	(−67.0, −16.8)	9	190
10	(−38.5, −6.0)	16	290	60	(−4.5, 18.0)	7	130
11	(−50.0, −27.6)	6	110	61	(−34.0, 26.4)	13	300
12	(−88.5, 1.8)	10	210	62	(−31.5, 10.8)	6	100
13	(−23.5, 18.0)	6	130	63	(9.5, −22.8)	9	220
14	(−80.0, 23.4)	16	330	64	(17.5, −3.6)	7	170
15	(−46.5, 24.0)	17	340	65	(81.5, −24.0)	5	70
16	(−56.5, 21.6)	8	130	66	(89.5, −15.6)	9	160
17	(−40.0, 13.2)	20	370	67	(43.5, −22.2)	11	260
18	(−16.5, 3.6)	7	130	68	(24.0, −18.0)	8	170
19	(−60.0, 10.2)	17	290	69	(9.5, 9.0)	6	80
20	(−66.5, 25.2)	6	120	70	(15.0, 6.0)	3	60
21	(−13.5, 16.8)	14	260	71	(82.0, 16.2)	6	120
22	(−10.0, 26.4)	20	430	72	(59.0, 24.6)	5	100
23	(−6.5, 11.4)	9	220	73	(31.5, 4.8)	3	50
24	(−50.0, 2.4)	9	200	74	(18.0, −9.6)	6	80
25	(−21.5, 9.6)	5	70	75	(18.5, 16.8)	11	200
26	(−33.5, 4.2)	8	140	76	(39.0, 12.6)	6	150
27	(−73.5, 10.2)	4	80	77	(−15.5, −21.0)	7	130
28	(66.5, −12.0)	12	250	78	(−6.0, 3.6)	4	110
29	(10.0, −6.6)	5	80	79	(−30.5, −21.0)	4	70
30	(43.5, −6.0)	6	100	80	(−22.5, 26.4)	5	110
31	(33.5, −8.4)	12	260	81	(−70.5, 18.6)	6	140
32	(50.0, −16.8)	12	200	82	(57.5, −8.4)	5	70
33	(23.5, −4.8)	7	130	83	(60.0, −22.2)	7	150
34	(76.5, −13.2)	14	250	84	(92.5, 18.0)	5	120
35	(80.0, −3.6)	14	270	85	(60.0, 13.8)	8	120
36	(51.5, −6.0)	7	140	86	(−30.0, −5.4)	3	50
37	(68.5, −20.4)	6	90	87	(−31.5, −15.0)	3	60
38	(1.5, 1.8)	5	90	88	(−52.5, −18.6)	6	110
39	(66.5, 18.0)	7	120	89	(2.0, 12.0)	5	130
40	(10.0, 23.4)	17	340	90	(10.5, −1.2)	4	100
41	(43.5, 24.0)	9	200	91	(84.0, 3.0)	10	220
42	(33.5, 21.6)	9	170	92	(39.0, 29.4)	4	110
43	(50.0, 13.2)	18	350	93	(−73.5, −19.8)	1	10
44	(70.0, 25.2)	11	210	94	(1.5, −28.2)	1	30
45	(73.5, 3.6)	6	150	95	(73.5, −26.4)	1	10
46	(30.0, 10.2)	6	130	96	(30.0, −19.8)	1	30
47	(23.5, 25.2)	3	80	97	(83.5, −18.6)	1	30
48	(76.5, 16.8)	12	290	98	(40.0, −27.6)	1	10
49	(80.0, 26.4)	10	230	99	(−24.5, 5.4)	1	30
50	(83.5, 11.4)	10	190	100	(19.0, 28.8)	1	20

Table 2. Space-robot capacity information.

Robot Number	Robot 1	Robot 2	Robot 3	Robot 4	Robot 5
Total capacity 5000	1200	950	950	950	950
Total capacity 6000	1500	1125	1125	1125	1125
Total capacity 7000	1800	1300	1300	1300	1300

Table 3. Comparison of task-planning paths for a space-robot cluster with a total energy of 5000.

Type	Robot	Consumption	Income	Number of Tasks	Income Rate
Multirobot optimal-path model	R ₁	945.53	10,370	525	65.34%
	R ₂	753.56			
	R ₃	738.72			
	R ₄	741.71			
	R ₅	752.41			
Multirobot shortest-path model	R ₁	934.61	6830	345	43.04%
	R ₂	658.22			
	R ₃	751.03			
	R ₄	692.13			
	R ₅	686.65			

Note: The total energy of robot cluster is 5000, the energy of each robot are [1200, 950, 950, 950, 950], the available energies are [960, 760, 760, 760, 760].

Table 4. Comparison of task-planning paths for a space-robot cluster with a total energy of 6000.

Type	Robot	Consumption	Income	Number of Tasks	Income Rate
Multirobot optimal-path model	R ₁	1184.51	12,640	632	79.65%
	R ₂	890.12			
	R ₃	888.59			
	R ₄	899.01			
	R ₅	896.14			
Multirobot shortest-path model	R ₁	1184.70	9160	466	57.84%
	R ₂	865.95			
	R ₃	845.31			
	R ₄	843.96			
	R ₅	880.99			

Note: The total energy of robot cluster is 6000, the energy of each robot are [1500, 1125, 1125, 1125, 1125], the available energies are [1200, 900, 900, 900, 900].

Table 5. Comparison of task-planning paths for a space-robot cluster with a total energy of 7000.

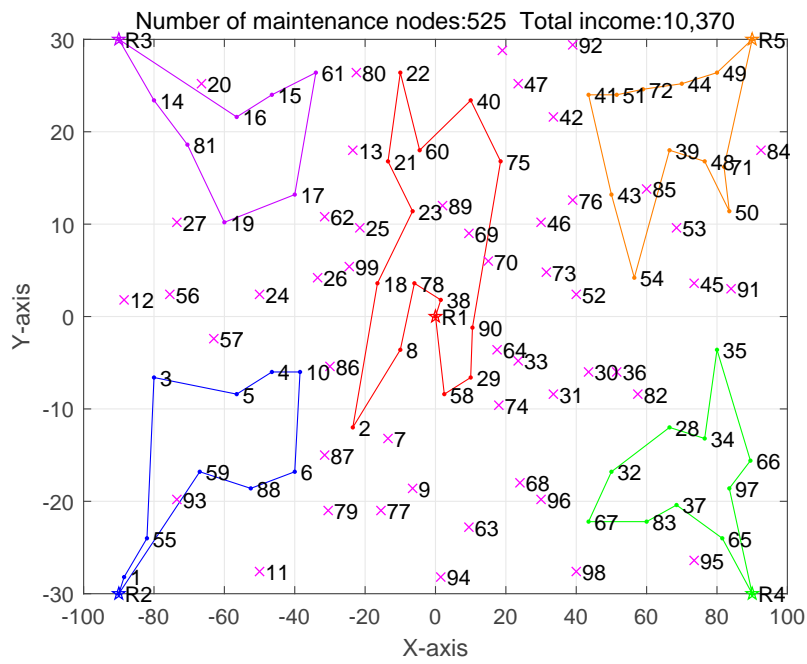
Type	Robot	Consumption	Income	Number of Tasks	Income Rate
Multirobot optimal-path model	R ₁	1433.22	14,540	730	91.62%
	R ₂	1035.71			
	R ₃	1024.98			
	R ₄	1036.98			
	R ₅	1021.90			
Multirobot shortest-path model	R ₁	1273.64	10,650	543	67.11%
	R ₂	961.64			
	R ₃	935.14			
	R ₄	1001.92			
	R ₅	1018.60			

Note: The total energy of the robot clusters is 7000, the energy of each robot is [1800, 1300, 1300, 1300, 1300]; the available energies are [1440, 1040, 1040, 1040, 1040].

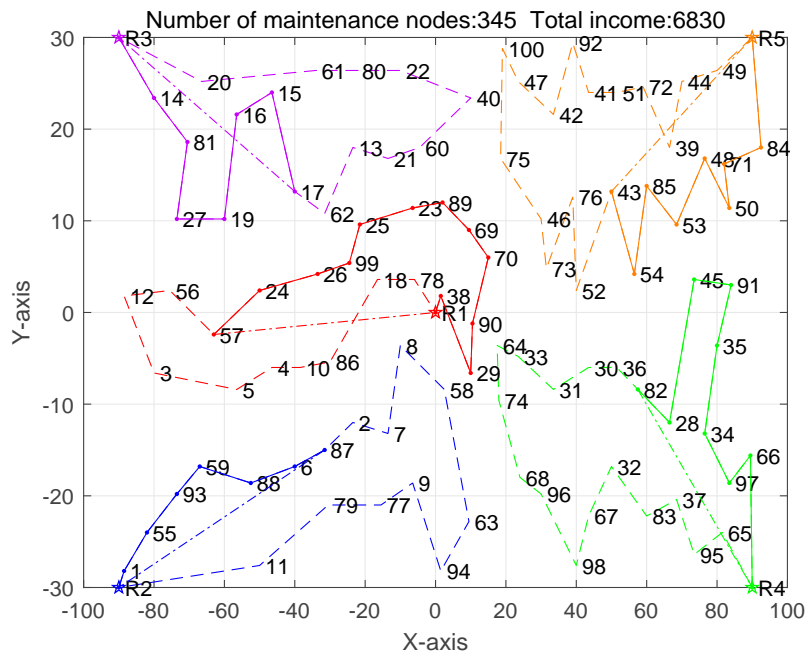
As can be seen from Table 3, the energy consumption of each space robot are basically close to its maximum of available energy, it has made full use of energy, but under the model of optimal path space-robot clustering task earnings significantly higher than that of the shortest-path model, which is about 1.38 times of the shortest-path model, the benefits of maintenance were obvious ascension; it can also be seen that the number of maintenance nodes in the optimal-path model is much higher than that in the shortest-path model, which is about 1.36 times that of the shortest-path model, by comparing the Income rate, it can be seen that the task return rate of the optimal path is also much higher than the shortest-path model. In conclusion, under the same energy constraints, the planning effect of the optimal-path model is better than that of the shortest-path model. Similarly, Tables 4 and 5 show the optimization results of the optimal path and the shortest path under the other two energies. By comparing Tables 3–5, the planning effect of the optimal-path model is better than that of the shortest-path model at different energy levels, which shows the superiority of the optimal-path model proposed in this paper.

In space, resource constraint is one of the main factors that affect the space robot, and they must maximize efficiency with limited resources. It can be clearly seen from Tables 3–5 that under certain resource conditions, the number of tasks completed by the optimal path, the total revenue and the task completion rate are obviously better than the shortest path. The total revenue of the optimal path is 42.33% more than the total revenue of the shortest path, and the number of tasks completed by the optimal path is 40.67% more than the average number of tasks completed by the shortest path. Therefore, the optimal-path model has obvious advantages in dealing with the task allocation problem of space-robot clusters with limited resources.

The specific maintenance routes of the space-robot cluster with energies of 5000, 6000 and 7000 are, respectively shown in Figures 3–5, the starting point of the optimized path is the location of each space charging pile. Figures 3–5 correspond to the paths in Tables 3–5, wherein star points represent the starting positions of the space robot, solid line loop in the optimal-path model represents the maintenance path of the space robot, and the path consumption of each robot corresponds to the corresponding table. The solid line and stippling line in the shortest path represent the maintenance path of the space robot, while the dotted line represents the position in the shortest path that the space robot cannot complete under the condition of limited energy. The stippling line is the route that the space robot can return to the charging pile position within a safe energy range. However, in some cases, because the points in the cluster are too scattered, the different results of the cluster will have a certain impact on the number of completed tasks. This experiment proves the feasibility of the optimal path.

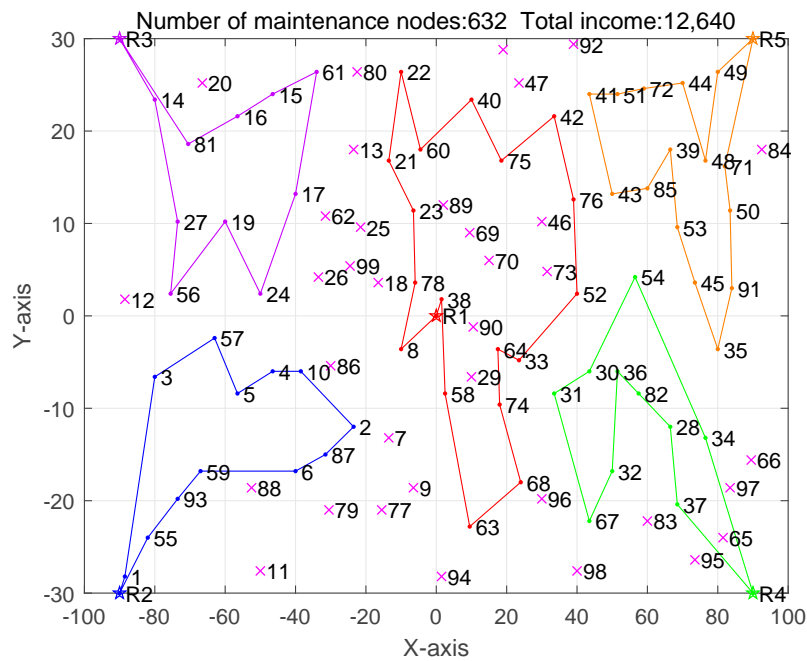


(a)

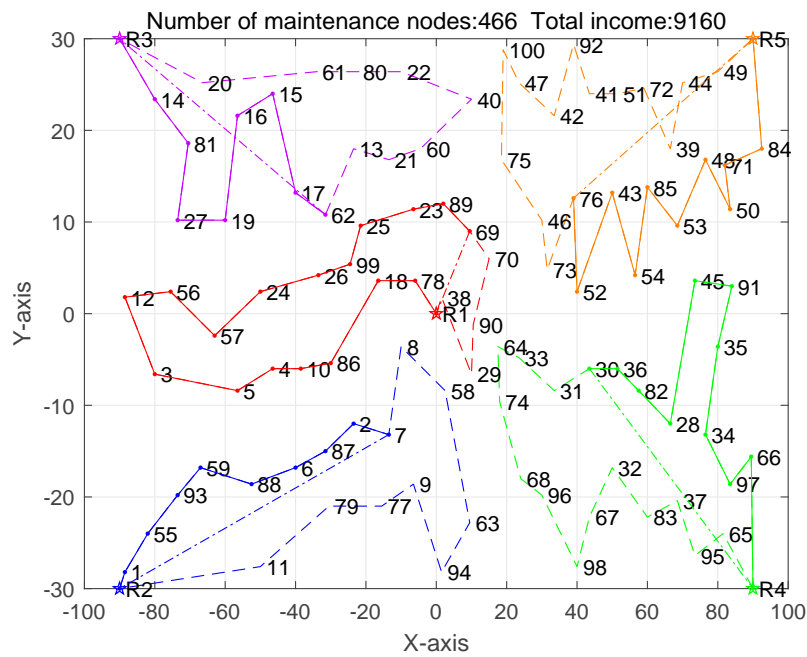


(b)

Figure 3. Comparison of maintenance routes when the space-robot cluster energy is 5000. (a) Optimal-path model; (b) shortest-path model.



(a)



(b)

Figure 4. Comparison of maintenance routes when the space-robot cluster energy is 6000. (a) Optimal-path model; (b) shortest-path model.

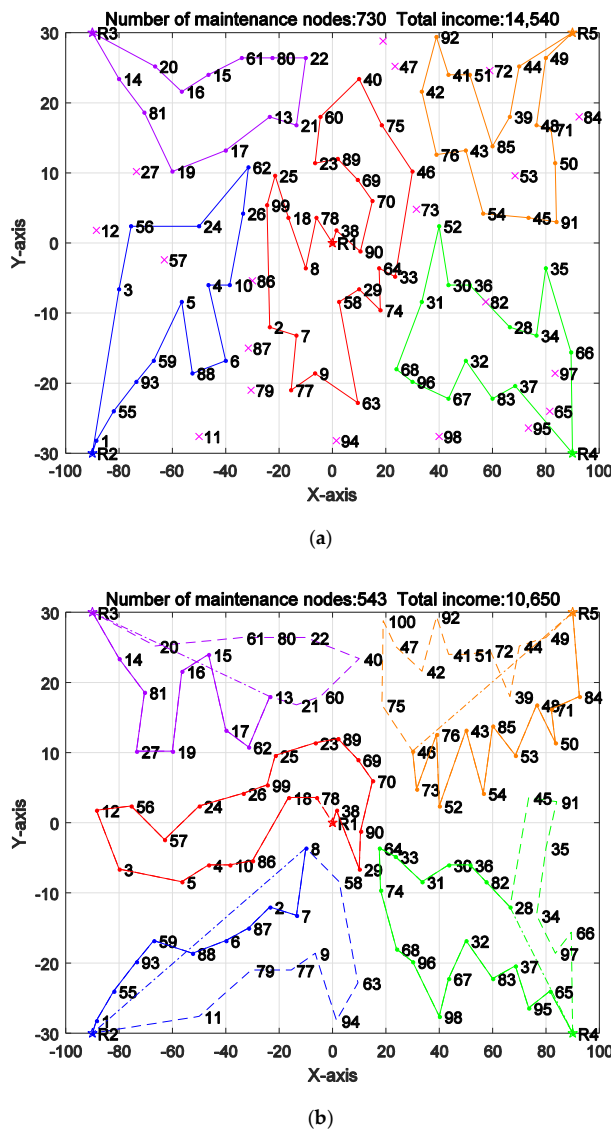


Figure 5. Comparison of maintenance routes when the space-robot cluster energy is 7000. (a) Optimal-path model; (b) shortest-path model.

5.2.2. Parameter Analysis

Since the modified differential evolution algorithm is very sensitive to parameters, different parameters will lead to different programming results for the same calculation. Taking the space-robot cluster at 6000 energy as an example, the optimization parameters are analyzed. The population number NP is selected according to experience, the commonly used range of $5D-10D$. In this paper, an adaptive mutation operator F is adopted, whose value can be adjusted adaptively. The crossover operator selects an appropriate value according to experience. In the modified differential evolution algorithm, the proportion of subpopulations λ (as $\lambda_1 = \lambda_2 = \lambda_3 = \lambda$) and ng have a great influence on the final optimization results. In order to obtain the optimal solution, this paper conducts a separate analysis on the influence of these two parameters, as shown in Table 6.

Table 6. The influence of λ and ng on the modified differential evolution algorithm.

Num	λ	ng	Maximum Income	Maintenance Points
1	0.10	20	11,660	585
2	0.15	20	12,030	606
3	0.20	20	12,240	605
4	0.25	20	12,640	632
5	0.25	10	12,180	609
6	0.25	30	11,640	580
7	0.25	40	11,450	565

We obtained by comparing multiple groups of experiments that when $\lambda = 0.25, ng = 20$, the effect of this experiment is the best. Different parameters may need to be set for different space maintenance problems. This paper provides a feasible parameter selection scheme.

5.3. Comparison Experiment of Modified Differential Evolution Algorithm

In order to verify the superiority of the algorithm in this experiment, taking the total energy of the space-robot cluster is 6000 as an example, the basic differential evolution algorithm (DE), the adaptive differential evolution algorithm (JDE), the multi-population integrated differential evolution algorithm (MPEDE) and the modified differential evolution algorithm (modified DE) are used, respectively to solve the optimal path of the maintenance task of the large SSPS. The test results are shown in Table 7.

Table 7. Comparison of different types of algorithms.

Algorithms	Objective Function Value	Total Income	Maintenance Points
DE	4300.27	5150	251
JDE [40]	8705.85	9590	468
MPEDE [41]	8906.75	9850	502
Modified DE	11,688.33	12,640	632

Some conclusions can be drawn by comparing the different types of algorithms in Table 7. The modified differential evolution algorithm is more effective than DE, JDE and MPEDE in the objective function value, the maximum profit and the number of maintenance points of the space-robot cluster. In terms of the total maintenance revenue of space-robot clusters, the modified differential evolution algorithm is 2.45 times that of DE, 1.32 times that of JDE and 1.28 times that of MPEDE. In the total maintenance points of space-robot cluster, the modified differential evolution algorithm is 2.52 times of DE, 1.35 times of JDE and 1.26 times of MPEDE. This shows that the modified difference algorithm proposed in this paper is better than the existing methods in solving the optimal-path model of spatial robot clusters.

Figure 6 shows the comparison of the maximum income of the space-robot cluster, Figure 7 shows the comparison of the maximum maintenance points. Figure 8 shows the comparison of the objective function value. At the beginning, due to the existence of the penalty coefficient, the objective function is negative, and it will be very large, which makes the graph display not obvious. In order to make it clear, we have truncated the vertical coordinate of Figure 8.

It can be seen from Figures 6–8 that the number of iterations of the modified differential evolution algorithm is significantly less than that of DE, JDE and MPEDE in solving the optimal solution of the maximum benefit, the number of maintenance points and the objective function value of the space-robot cluster. The convergence effect of the modified differential evolution algorithm is better than that of other algorithms.

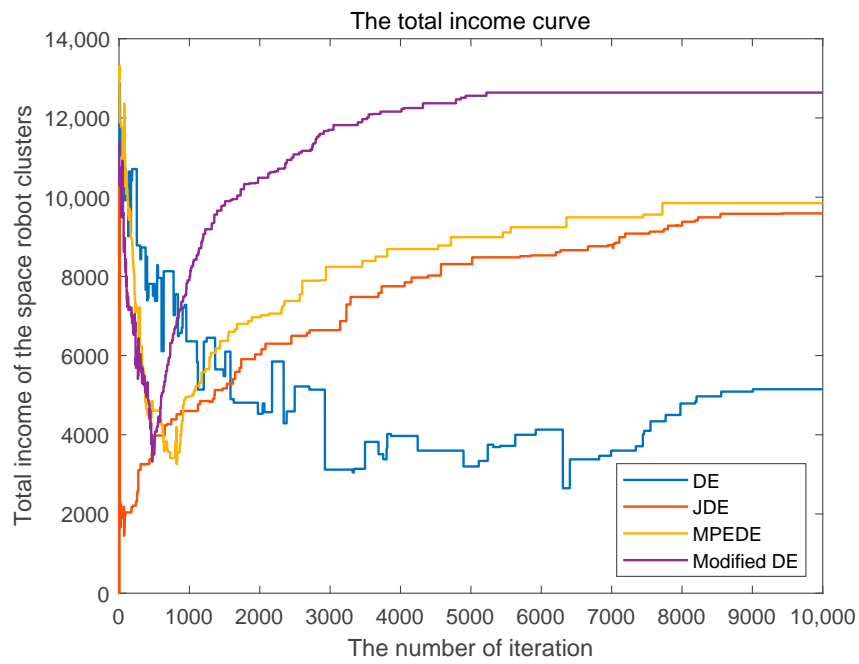


Figure 6. Total income when the space-robot cluster energy is 6000.

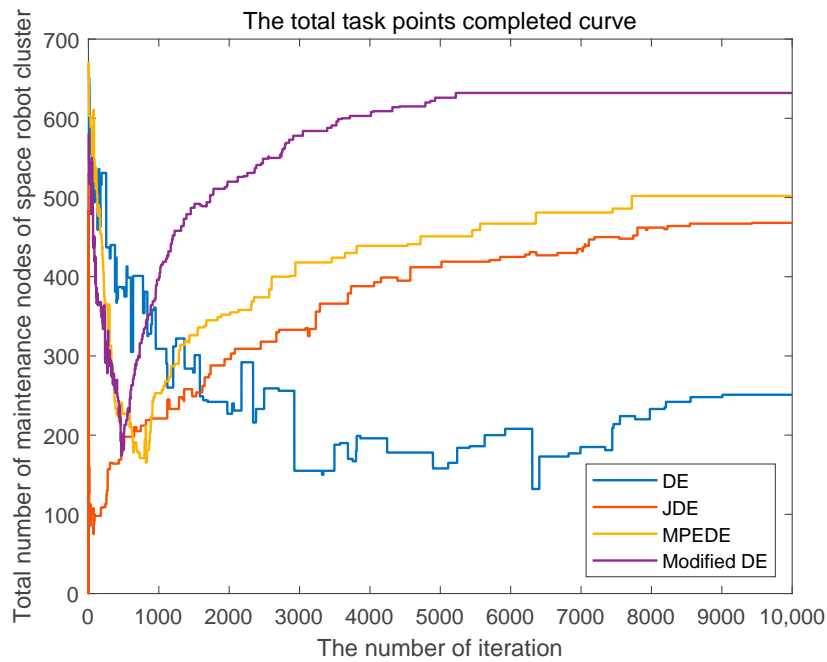


Figure 7. Total task points completed when the space-robot cluster energy is 6000.

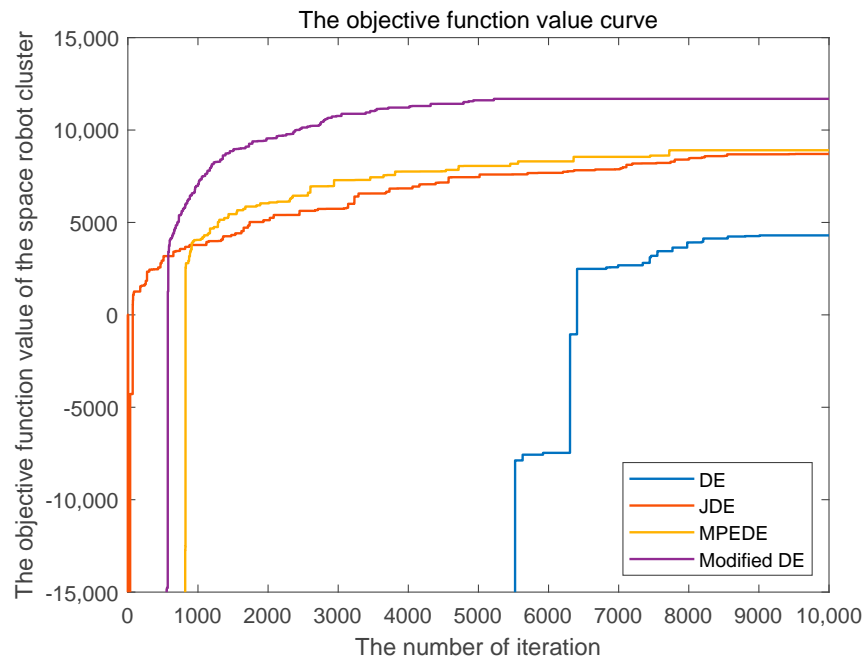


Figure 8. Value of the objective function when the space-robot cluster energy is 6000.

In this paper, the variation factor of differential evolution algorithm is improved, the diversity of individuals is maintained in the early stage, the mutation operator is reduced in the later stage and the probability of searching the global optimal solution is increased. The combination of roulette and multi-neighborhood operations is used to ensure the optimal value of the population and avoid falling into the local optimal; the convergence of the model is accelerated by the de-crossing strategy; the multi-population strategy-integration method is used to obtain more computational resources in the evolution of better mutation strategies. Therefore, the modified differential evolution algorithm proposed in this paper has the advantages of global optimality and fast convergence speed, which improves the convergence speed and convergence effect of the algorithm, and the overall performance is better than other existing algorithms.

For the optimal path problem of space robots for maintenance tasks of large space solar-power stations, in order to better prove the performance of the modified DE proposed in this paper, we take the space-robot cluster with a total energy of 6000 as an example, the genetic algorithm (GA), ant colony optimization algorithm (ACO), artificial bee colony algorithm (ABC) and modified DE are, respectively used to solve the optimal path of maintenance task of large space solar-power station. The test results are shown in Table 8.

Table 8. Comparison of other algorithms.

Algorithms	Objective Function Value	Maximum Income	Maintenance Points
GA	5794.99	6470	321
ACO	7049.01	7790	396
ABC	10,246.18	11,200	561
modified DE	11,688.33	12,640	632

By comparing the different optimization algorithms in Table 8, we can see that in the space maintenance task, the modified differential evolution algorithm is better than the GA, ACO and ABC in solving the optimal solution of the maximum benefit, the number of maintenance points and the objective function value of the space-robot cluster. Therefore, this paper chooses to use the modified DE for the optimal path planning of the space robot, so that it can better complete the space maintenance task.

6. Conclusions

The on-orbit maintenance task planning of large SSPS is an important research hotspot and the use of space-robot clusters to complete space maintenance operations is a development trend. Energy in space is one of the important factors to be considered. In order to solve this problem, we have modeled and analyzed the space-robot cluster problem. The optimal-path model of the space-robot cluster proposed in this paper improves the task planning effect of the space-robot cluster. The superiority of modeling is verified by comparing with the shortest-path model. Clustering operations on target maintenance nodes can greatly reduce the difficulty and time loss of task planning for space-robot clusters and effectively improve the efficiency of space robot task allocation. Aiming at the defects of slow convergence speed and easy to fall into the local optimum in the later stage of differential evolution algorithm, this paper improves the convergence speed and convergence effect of the algorithm through a series of improvement measures and proves that the modified DE can solve the optimal path problem of space-robot cluster more effectively through simulation experiments. This article provides an efficient algorithm to solve the problem of space-robot cluster task-planning for space maintenance operations; it will also have very good benefits for the maintenance problem of larger space agencies in the future.

Future research can integrate other realistic constraints in the space robot task model, such as considering the space robot's time window, battery life, network communication, attitude orbit control, etc. On the other hand, in the process of task planning for the space-robot cluster, the space environment and the actual situation of the space robot are considered, dynamic task assignment is performed and the problem of secondary assignment of tasks is considered.

Author Contributions: P.X. and Q.L. provided conceptualization and resources. P.X. did the investigation, methodology and the validation, wrote the software. Q.L. established the computational model, discussed the results and wrote the study. H.J. and F.C. did the formal analysis and reviewed and edited the manuscript. All authors have read and agreed to the published version of the manuscript.

Funding: This research was funded by the National Natural Science Foundation of China, Grant Number 61673010.

Conflicts of Interest: The authors declare no conflict of interest.

References

1. Glaser, P.E. Power from the Sun: Its Future. *Science* **1968**, *162*, 857–861. [[CrossRef](#)] [[PubMed](#)]
2. Oegerle, W.R.; Purves, L.; Budinoff, J.; Moe, R.V.; Carnahan, T.M.; Evans, D.C.; Kim, C.K. Concept for a large scalable space telescope: In-space assembly. *Proc. SPIE* **2006**, *6265*, 62652C.
3. Yoshida, K.; Hashizume, K.; Abiko, S. Zero reaction maneuver: Flight validation with ETS-VII space robot and extension to kinematically redundant arm. In Proceedings of the 2001 ICRA. IEEE International Conference on Robotics and Automation (Cat. No. 01CH37164), Seoul, Korea, 21–26 May 2001; pp. 441–446.
4. Zemlyakov, S.D.; Rutkovskii, V.Y.; Sukhanov, V.M.O. Some questions of control of the robotized in-orbit assembly of large space structures. *Autom. Remote Control* **2006**, *67*, 1215–1227. [[CrossRef](#)]
5. Lee, D. Resource-based task allocation for multi-robot systems. *Robot. Auton. Syst.* **2018**, *103*, 151–161. [[CrossRef](#)]
6. Kanakia, A.; Touri, B.; Correll, N. Modeling multi-robot task allocation with limited information as global game. *Swarm Intell.* **2016**, *10*, 147–160. [[CrossRef](#)]
7. Yan, Z.; Jouandeau, N.; Cherif, A.A. A Survey and Analysis of Multi-Robot Coordination. *Int. J. Adv. Robot. Syst.* **2013**, *10*, 399. [[CrossRef](#)]
8. Badreldin, M.; Hussein, A.; Khamis, A. A comparative study between optimization and market-based approaches to multi-robot task allocation. *Adv. Artif. Intell.* **2013**, *2013*, 12. [[CrossRef](#)]
9. Vig, L.; Adams, J.A. Market-Based Multi-robot Coalition Formation. In *Proceedings of the Distributed Autonomous Robotic Systems*; Springer: Tokyo, Japan, 2006; pp. 227–236.
10. Yin, P.; Yu, S.; Wang, P.; Wang, Y. Multi-objective task allocation in distributed computing systems by hybrid particle swarm optimization. *Appl. Math. Comput.* **2007**, *184*, 407–420. [[CrossRef](#)]

11. Carter, A.E.; Ragsdale, C.T. A new approach to solving the multiple traveling salesperson problem using genetic algorithms. *Eur. J. Oper. Res.* **2006**, *175*, 246–257. [[CrossRef](#)]
12. Tan, Y.; Zheng, Z. Research Advance in Swarm Robotics. *Def. Technol.* **2013**, *9*, 18–39. [[CrossRef](#)]
13. Zhang, H.; Zhang, Q.; Ma, L.; Zhang, Z.; Liu, Y. A hybrid ant colony optimization algorithm for a multi-objective vehicle routing problem with flexible time windows. *Inf. Sci.* **2019**, *490*, 166–190. [[CrossRef](#)]
14. Moisiadis, V.; Tsolakis, N.; Katikaridis, D.; Sørensen, C.G.; Bochtis, D. Mobile Robotics in Agricultural Operations: A Narrative Review on Planning Aspects. *Appl. Sci.* **2020**, *10*, 3453. [[CrossRef](#)]
15. Zhen, Z.; Xing, D.; Gao, C. Cooperative search-attack mission planning for multi-UAV based on intelligent self-organized algorithm. *Aerosp. Sci. Technol.* **2018**, *76*, 402–411. [[CrossRef](#)]
16. Han, Y.; Li, D.; Chen, J.; Yang, X.; Hu, Y.; Zhang, G. A Multi-Robots Task Allocation Algorithm Based on Relevance and Ability with Group Collaboration. *Int. J. Intell. Eng. Syst.* **2010**, *3*, 33–41. [[CrossRef](#)]
17. Yao, W.; Qi, N.; Wan, N.; Liu, Y. An iterative strategy for task assignment and path planning of distributed multiple unmanned aerial vehicles. *Aerosp. Sci. Technol.* **2019**, *86*, 455–464. [[CrossRef](#)]
18. Schwarzrock, J.; Zacarias, I.; Bazzan, A.L.C.; Fernandes, R.Q.D.A.; Moreira, L.H.; De Freitas, E.P. Solving task allocation problem in multi Unmanned Aerial Vehicles systems using Swarm intelligence. *Eng. Appl. Artif. Intell.* **2018**, *72*, 10–20. [[CrossRef](#)]
19. Gerkey, B.P.; Mataric, M.J. A Formal Analysis and Taxonomy of Task Allocation in Multi-Robot Systems. *Int. J. Robot. Res.* **2004**, *23*, 939–954. [[CrossRef](#)]
20. Edison, E.; Shima, T. Integrated task assignment and path optimization for cooperating uninhabited aerial vehicles using genetic algorithms. *Comput. Oper. Res.* **2011**, *38*, 340–356. [[CrossRef](#)]
21. Lee, M.; Chen, B.; Lu, W. Failure-Robot Path Complementation for Robot Swarm Mission Planning. *Appl. Sci.* **2019**, *9*, 3756. [[CrossRef](#)]
22. Ye, F.; Chen, J.; Tian, Y.; Jiang, T. Cooperative Multiple Task Assignment of Heterogeneous UAVs Using a Modified Genetic Algorithm with Multi-type-gene Chromosome Encoding Strategy. *J. Intell. Robot. Syst.* **2020**, 1–13. [[CrossRef](#)]
23. Deng, Q.; Yu, J.; Wang, N. Cooperative task assignment of multiple heterogeneous unmanned aerial vehicles using a modified genetic algorithm with multi-type genes. *Chin. J. Aeronaut.* **2013**, *26*, 1238–1250. [[CrossRef](#)]
24. Kitjacharoenchai, P.; Ventresca, M.; Moshref-Javadi, M.; Lee, S.; Tanchoco, J.M.A.; Brunese, P.A. Multiple traveling salesman problem with drones: Mathematical model and heuristic approach. *Comput. Ind. Eng.* **2019**, *129*, 14–30. [[CrossRef](#)]
25. Hussein, A.; Marinplaza, P.; Garcia, F.; Armingol, J.M. Hybrid Optimization-Based Approach for Multiple Intelligent Vehicles Requests Allocation. *J. Adv. Transp.* **2018**, *2018*, 2493401. [[CrossRef](#)]
26. Zheng, T.; Li, J. Multi-robot task allocation and scheduling based on fish swarm algorithm. In Proceedings of the 2010 8th World Congress on Intelligent Control and Automation, Jinan, China, 7–9 July 2010; pp. 6681–6685.
27. Chen, X.; Zhang, P.; Du, G.; Li, F. Ant Colony Optimization Based Memetic Algorithm to Solve Bi-Objective Multiple Traveling Salesmen Problem for Multi-Robot Systems. *IEEE Access* **2018**, *6*, 21745–21757. [[CrossRef](#)]
28. Khan, I.; Maiti, M.K. A swap sequence based Artificial Bee Colony algorithm for Traveling Salesman Problem. *Swarm Evol. Comput.* **2019**, *44*, 428–438. [[CrossRef](#)]
29. Trigui, S.; Cheikhrouhou, O.; Koubaa, A.; Baroudi, U.; Youssef, H. FL-MTSP: A fuzzy logic approach to solve the multi-objective multiple traveling salesman problem for multi-robot systems. *Soft Comput.* **2017**, *21*, 7351–7362. [[CrossRef](#)]
30. Mahi, M.; Baykan, O.K.; Kodaz, H. A new hybrid method based on Particle Swarm Optimization, Ant Colony Optimization and 3-Opt algorithms for Traveling Salesman Problem. *Appl. Soft Comput.* **2015**, *30*, 484–490. [[CrossRef](#)]
31. Cheng, Z.; Hou, X.; Zhang, X.; Zhou, L.; Guo, J.; Song, C. In-orbit assembly mission for the Space Solar Power Station. *Acta Astronaut.* **2016**, *129*, 299–308. [[CrossRef](#)]
32. Li, X.; Duan, B.; Song, L.; Yang, Y.; Zhang, Y.; Wang, D. A new concept of space solar power satellite. *Acta Astronaut.* **2017**, *136*, 182–189. [[CrossRef](#)]
33. Storn, R.; Price, K. Differential Evolution—A Simple and Efficient Heuristic for global Optimization over Continuous Spaces. *J. Glob. Optim.* **1997**, *11*, 341–359. [[CrossRef](#)]
34. Rodriguez, A.; Laio, A. Clustering by fast search and find of density peaks. *Science* **2014**, *344*, 1492–1496. [[CrossRef](#)]

35. Hartigan, J.A.; Wong, M.A. A K-Means Clustering Algorithm. *J. R. Stat. Soc. Ser. C Appl. Stat.* **1979**, *28*, 100–108.
36. Xiao, P.; Ju, H.; Li, Q.; Xu, H.; Lu, C. Task Planning of Space Maintenance Robot Using Modified Clustering Method. *IEEE Access* **2020**, *8*, 45618–45626. [[CrossRef](#)]
37. Fraley, C.; Raftery, A.E. Model-Based Clustering, Discriminant Analysis, and Density Estimation. *J. Am. Stat. Assoc.* **2002**, *97*, 611–631. [[CrossRef](#)]
38. Rahnamayan, S.; Tizhoosh, H.R.; Salama, M.M.A. A novel population initialization method for accelerating evolutionary algorithms. *Comput. Math. Appl.* **2007**, *53*, 1605–1614. [[CrossRef](#)]
39. Zhang, J.; Sanderson, A.C. JADE: Adaptive Differential Evolution with Optional External Archive. *IEEE Trans. Evol. Comput.* **2009**, *13*, 945–958. [[CrossRef](#)]
40. Brest, J.; Greiner, S.; Boskovic, B.; Mernik, M.; Zumer, V. Self-Adapting Control Parameters in Differential Evolution: A Comparative Study on Numerical Benchmark Problems. *IEEE Trans. Evol. Comput.* **2006**, *10*, 646–657. [[CrossRef](#)]
41. Wu, G.; Mallipeddi, R.; Suganthan, P.N.; Wang, R.; Chen, H. Differential evolution with multi-population based ensemble of mutation strategies. *Inf. Sci.* **2016**, *329*, 329–345. [[CrossRef](#)]
42. Tong, L.; Dong, M.; Jing, C. An improved multi-population ensemble differential evolution. *Neurocomputing* **2018**, *290*, 130–147. [[CrossRef](#)]



© 2020 by the authors. Licensee MDPI, Basel, Switzerland. This article is an open access article distributed under the terms and conditions of the Creative Commons Attribution (CC BY) license (<http://creativecommons.org/licenses/by/4.0/>).