Hindawi
Modelling and Simulation in Engineering
Volume 2021, Article ID 6644339, 15 pages
https://doi.org/10.1155/2021/6644339

Research Article

Deployment of On-Orbit Service Vehicles Using a Fuzzy Adaptive Particle Swarm Optimization Algorithm

Yaxiong Li, Xinglong Sun, Xinxue Liu, Jian Wu, and Qingguo Liu

Xi'an High-Tech Institute, Xi'an 710025, China

Correspondence should be addressed to Xinglong Sun; 1229594814@qq.com

Received 25 October 2020; Revised 9 April 2021; Accepted 5 June 2021; Published 21 June 2021

Academic Editor: Feng Xiao

Copyright © 2021 Yaxiong Li et al. This is an open access article distributed under the Creative Commons Attribution License, which permits unrestricted use, distribution, and reproduction in any medium, provided the original work is properly cited.

On the basis that satellites given fixed count and orbit elements can be served in bounded time when an on-orbit serving mission order is set at any uncertain time in a given time interval, the deployment of on-orbit service vehicle (OSV) serving satellites becomes a complex multiple nested optimization problem, and the essence of deployment is to determine the count and orbit elements of OSVs. In consideration of the characteristics of this deployment problem, we propose a fuzzy adaptive particle swarm optimization (FAPSO) algorithm to solve this problem. First, on the basis of double pulse rendezvous hypothesis, a transfer optimization model of a single OSV serving multiple satellites is established based on genetic algorithm (GA), and this is used to compute the indexes of the subsequent two optimization models. Second, an assignment optimization model of OSVs is established based on the discrete particle swarm optimization (DPSO) algorithm, laying the foundation of the next optimization model. Finally, the FAPSO algorithm, which improves the performance of PSO algorithm by adjusting the inertia weight, is proposed to solve the deployment problem of multiple OSVs. The simulation results demonstrate that all optimization models in this study are feasible, and the FAPSO algorithm, which has a better convergence result than that obtained using the other optimization algorithms, can effectively solve the deployment problem of OSVs.

1. Introduction

The on-orbit service of a spacecraft is a type of space operation that humans, robots, or both extend lives and improve their abilities in performing tasks of various spacecraft [1–3]. The main tasks of on-orbit service vehicles (OSVs) include on-orbit assembly, on-orbit maintenance, and logistic support [4]. As humans explore deeper in space, more and more diverse satellites are deployed in space. The main reasons why satellites are abandoned are that some components are broken and the fuel to maintain their orbits is not enough. The on-orbit service of a spacecraft is an effective method to reduce the waste of satellite resource and avoid producing space junk. Using OSVs to extend the lives of satellites deployed and improve the ability to perform tasks would be an important research direction.

The basic work of deployment of OSVs includes transfer orbit optimization and assignment optimization. In the existing literature, direct methods [5, 6], indirect methods [7], and hybrid methods [8] are used to study the optimization of transfer trajectories of different types of spacecraft. OSV transfer orbit is one kind of transfer orbits. Although the direct methods suffer from the lack of high precision, its computation is efficient. Considering the deployment and assignment optimization need to calculate a lot of transfer orbits under different states, we use direct method to compute transfer orbits. The literature related to assignment optimization and the deployments of OSVs are as follows. A novel approach for the mission planning of on-orbit servicing such as visual inspection is proposed by Daneshjou et al. [9]. The on-orbit servicing task allocation is solved by a hybrid discrete particle swarm optimization algorithm which is proposed by Zhang and Zhang [10]. Pradeepmon et al. [11] solved quadratic assignment problems by using discrete particle swarm optimization (DPSO) algorithm.

Deployment of OSVs serving satellites, which is a complex multiple nested optimization problem, is almost not systematically studied in the existing literatures. Deployment of OSVs is the application of deployment problem. The optimization methods of other related deployment problems

mainly include some intelligent algorithms. Tesarova & Vokalova [12] used genetic algorithm (GA) to deploy radio stations. Singh et al. [13] used particle swarm optimization (PSO) algorithm to deploy directional sensors. Ahmadian et al. [14] solved the large dimension electric distribution network problem by using a hybrid algorithm combining PSO and TS (Tabu search) algorithm (PSO+TS). Zhang et al. [15] obtained optimal deployment of flow monitors based on a hybrid algorithm combining GA and SA (simulated annealing) algorithm (GA+SA). The convergences of these intelligent algorithms are not good to compute complex multiple nested optimization problem. Thus, we propose a fuzzy adaptive particle swarm optimization (FAPSO) algorithm to improve the convergence of the results by balancing global and local search capability of PSO algorithm. The study background of the present study is that multiple satellites given fixed count and orbit elements need to be maintained by multiple OSVs in bounded time when an on-orbit serving mission order is set at any uncertain time t in a given time interval T (on-orbit mission). The main content of the present study consists of four parts (Sections 2–5). The framework of the present study is shown in Figure 1.

Part A (Section 2): on the basis of the double pulse rendezvous hypothesis and taking the maximum weight of remaining fuel as the optimization index (optimization index A), a transfer optimization model of single OSV serving multiple satellites is established based on the GA, which is used to calculate the indexes of the subsequent optimization models.

Part B (Section 3): at the time t of an on-orbit serving mission order set and taking the maximum value of both the count of satellites that can be served and the minimum value of a single OSV's optimization index A among all OSVs as the optimization index (optimization index B), an assignment optimization model of multiple OSVs serving multiple satellites is established based on DPSO algorithm.

Part C (Section 4): first of all, taking the time t in a given time interval T as the optimization variable and taking the minimum value of optimization index B as the optimization index (optimization index C), the optimization model to determine whether OSVs given initial count and orbit elements can achieve on-orbit mission or not is established based on the pattern search method and this is used to calculate the optimization index of the deployment model. Then, given the count of OSVs and taking the maximum value of optimization index C as the optimization index (optimization index D), the deployment optimization model of multiple OSVs serving multiple satellites is established based on the fuzzy adaptive particle swarm optimization (FAPSO) algorithm.

Part D (Section 5): the feasibility of all optimization models in the present study is verified through comparison with other intelligent algorithms, and the FAPSO algorithm is more efficient in solving the deployment problem of OSVs.

2. Transfer Orbit Optimization Model

A single OSV can serve multiple satellites. The procedure of the double pulse rendezvous method is that a single OSV manoeuvres because of the 1st pulse thrust at some time which is called manoeuvre time, flies without thrust for some time, and manoeuvres because of the 2nd pulse thrust to get the same velocity as the first satellite at some time which is called rendezvous time when both the OSV and the first satellite reach the same position. The OSV begins to serve the first satellite after the double pulse rendezvous procedure. After serving the first satellite, the OSV serves other satellites in the same way as it served the first satellite.

The procedure of the OSV serving the 1st satellite by double pulse rendezvous method is shown in Figure 2. The orbit of the OSV is original orbit, and the orbit of a satellite is target orbit. The position of the OSV at the manoeuvre time is point P_1, and the position of the OSV at the rendezvous time is point P_2. When the OSV manoeuvres from point P_1 on the original orbit to point P_2 on target orbit in Δt seconds to serve the satellite, $\mathbf{r}_1$ and $\mathbf{v}_1$ are its position vector and velocity vector, respectively, at point P_1, and $\mathbf{r}_2$ and $\mathbf{v}_2$ are its position vector and velocity vector, respectively, at point P_2. The determination of the 1st velocity increment $\Delta\mathbf{v}_{11}$ (Formula (1)) of the OSV serving the 1st satellite at point P_1 on the original orbit is called the Lambert problem, and the method to solve the Lambert problem is studied by Schumacher et al. [16]. The 2nd velocity increment $\Delta\mathbf{v}_{12}$ of the OSV serving the 1st satellite at point P_2 on the target orbit is written as Formula (2). After serving the 1st satellite for a certain time, the OSV begins to serve the 2nd satellite in the same procedure as it served the 1st satellite.

$$\Delta\mathbf{v}_{11} = \mathbf{v}_1 - \mathbf{v}_0, \tag{1}$$

$$\Delta\mathbf{v}_{12} = \mathbf{v}_3 - \mathbf{v}_2. \tag{2}$$

In Formula (1) and Formula (2), $\mathbf{v}_0$ is the velocity of the OSV at point P_1 on the original orbit, $\mathbf{v}_1$ is the velocity of the OSV at point P_1 on the transfer orbit after pushing by the 1st pulse thrust, $\mathbf{v}_2$ is the velocity of the OSV at point P_2 on the transfer orbit, and $\mathbf{v}_3$ is the velocity of the OSV at point P_2 on the target orbit after pushing by the 2nd pulse thrust.

Given the sequence of serving satellites and taking the maximum weight of the remaining fuel as the optimization index, the transfer orbit optimization model of a single OSV serving multiple satellites is established by the GA. The manoeuvre and rendezvous times are searched globally using the GA. Optimization index A can be obtained by calculating the velocity increments of the procedures of an OSV serving multiple satellites.

2.1. Mathematical Model. From the viewpoint of energy, problems correlated to multiple OSVs serving multiple satellites are investigated in the present study. The weight of a single OSV includes the weight of the fuel it carries and the weight of its other parts.

$$m = m_{\text{fuel}} + m_{\text{else}}. \tag{3}$$

In Formula (3), m is the weight of a single OSV, m_{fuel} is the weight of the fuel carried by the OSV, and m_{else} is the weight of the other parts.

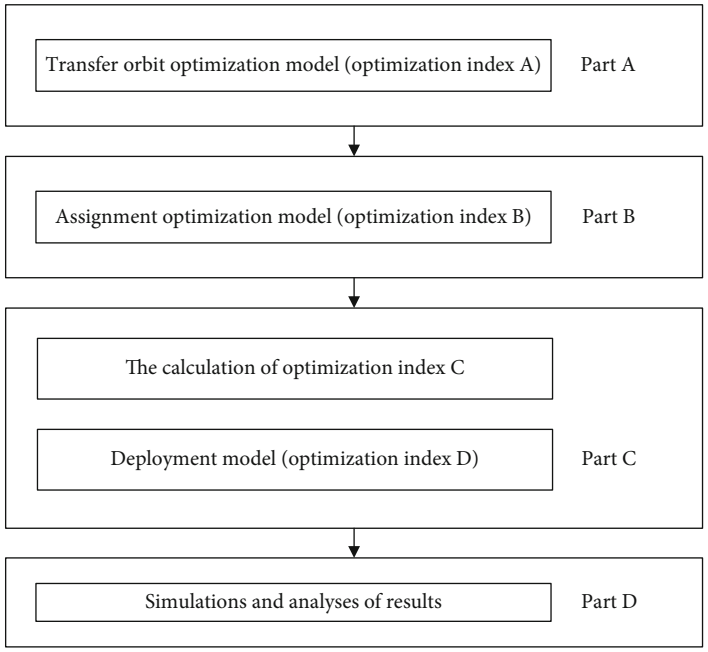

Figure 1: Framework of the present study.

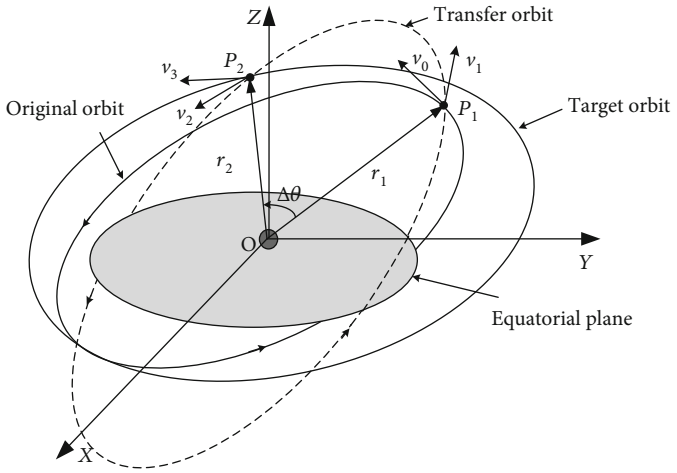

Figure 2: The procedure of the double pulse rendezvous method.

Given the sequence of serving satellites, the mathematical model can be written as

$$\max J = m_{\text{remain}}$$
$$\text{s.t.} \begin{cases} t_1 < t_2 < t_3 < t_4 < \cdots < t_{2n-1} < t_{2n} < T_{\max} - t_{\text{serve}} \\ t_3 - t_2 \geq t_{\text{serve}}, \cdots, t_{2n-1} - t_{2n-2} \geq t_{\text{serve}} \\ \sum_{i=1}^{n} \Delta m_i \leq m_{\text{fuel}} - m_{\text{keep}}. \end{cases} \tag{4}$$

In Formula (4), J is the optimization index A, m_{remain} is the weight of the remaining fuel after the on-orbit mission is achieved, $t_1, t_3, \cdots, t_{2n-1}$ are the manoeuvre times, $t_2, t_4, \cdots, t_{2n}$ are the rendezvous times, $T_{\max}$ is the maximum time interval for finishing the on-orbit mission, t_{serve} is the time interval for the OSV serving one satellite, n is the count of the transfer orbit segments (the i^{th} segment corresponding to the i^{th} satellite served by the OSV), Δm_i is the weight of the consumed fuel in the i^{th} segment, and m_{keep} is the weight of fuel used to keep the orbit. m_{remain} can be written as Formula (5) and Δm_i can be written as Formula (6).

$$m_{\text{remain}} = m_{\text{fuel}} - \sum_{i=1}^{k} \Delta m_i, k \leq K, \tag{5}$$

$$\Delta m_i = \left\{ m_{i-1} \left[1 - \exp\left(-\frac{\Delta v_{i1}}{g_o I_{\mathrm{sp}}} \right) \right] \right\} \left[1 - \exp\left(-\frac{\Delta v_{i2}}{g_o I_{sp}} \right) \right], i \le K. \tag{6}$$

In Formulas (5) and (6), k is the count of satellites served, K is the count of satellites that single OSV can serve, m_{i-1} is the weight of the remaining fuel after a single OSV serves $i-1$ satellites, I_{sp} is the specific impulse of the OSV, g_o is the acceleration of the earth, m_{i-1} is the weight of the remaining fuel after the OSV serves the $(i-1)^{\mathrm{th}}$ satellite, Δv_{i1} is the 1^{st} velocity increment of the OSV serving the i^{th} satellite at the manoeuvre time, and Δv_{i2} is the 2^{nd} velocity increment of the OSV serving the i^{th} satellite at the rendezvous time.

2.2. Optimization Model. The GA is an intelligence algorithm which is widely used in many fields [17–19]. Taking the maximum value of the remaining fuel weight (max m_{remain}) as the optimization index, the transfer optimization model of a single OSV serving multiple satellites is established based on the GA. The implementation steps are as follows.

Step 1. In order to meet the constraint conditions in Formula (4), the variables $t_{10}, t_{21}, \cdots, t_{2n(2n-1)}$ are written as

$$\begin{cases} t_{2i(2i-1)} = t_{2i} - t_{2i-1} \ge 3600, & i = 1, 2, \cdots, n, \\ t_{(2i-1)(2i-2)} = t_{2i-1} - t_{2i-2} \ge 0, & i = 1, 2, \cdots, n. \end{cases} \tag{7}$$

In Formula (7), n is the count of the transfer orbit segments. Furthermore, $t_{10}, t_{21}, \cdots, t_{2n(2n-1)}$ are taken as genes encoded to a chromosome (Figure 3). The initial populations are randomly generated. The population size M_t, the crossover probability P_c, the mutation probability P_m, and the maximum number of iterations T_A are set.

Step 2. The object function of Formula (4) is taken as the fitness function of the GA. Each chromosome has a fitness function value. The fitness function values of all chromosomes in the current generation are calculated.

Step 3. The chromosomes are selected by means of roulette wheel selection. Some genes on two different chromosomes reciprocally cross according to the crossover probability. Furthermore, some genes mutate according to the mutation probability.

Step 4. Steps 2 and 3 are repeatedly executed until the GA converged ($|J(R_i) - J(R_{i-1})| \le 10^{-6}$) or the maximum number of iterations is reached.

Given the sequence of serving satellites, the maximum J could be obtained through the GA.

3. Assignment Optimization Model

3.1. Mathematical Model

t_{10}	t_{21}	…	$t_{2n(2n-1)}$

Figure 3: Coding of the chromosome.

Assumption 1. There are N OSVs that are remarked as O = (O1, **O**2,⋯,ON), there are M satellites that are remarked as S = (S1, S2,⋯,SM), a single OSV serves K satellites at most, and a single satellite can be served once at most. The assignment matrix $\mathbf{X}$ is written as

$$x_{ij} = \begin{cases} 1, & \mathrm{O}i \text{ serving S}j, \\ 0, & \mathrm{O}i \text{ not serving S}j. \end{cases} \tag{8}$$

In Formula (8), $i = 1, 2, \cdots, N$, $j = 1, 2, \cdots, M$. The mathematical model is written as

$$\max f(\mathbf{X}) = \sum_{i=1}^{N} \sum_{j=1}^{M} x_{ij} + \frac{\min m_{\mathrm{remain}}^{s}}{m}$$

$$\text{s.t.} \begin{cases} \sum_{j=1}^{M} x_{ij} \le K, \forall i = 1, 2, \cdots, N \\ \sum_{i=1}^{N} x_{ij} \le 1, \forall j = 1, 2, \cdots, M \\ m_{\mathrm{remain}}^{s} \ge m_{\mathrm{keep}}, \forall s = 1, 2, \cdots, N. \end{cases} \tag{9}$$

In Formula (9), $\max f(\mathbf{X})$ is the optimization index B. Furthermore, K is the count of satellites that a single OSV can serve, m is the weight of a single OSV, m_{remain}^{s} is the weight of the remaining fuel after the s^{th} OSV serves the satellites, and m_{keep} is the weight of the fuel used to keep the orbit.

Optimization index B includes two parts.

The first part (optimization index B_1) is written as the formula $\sum_{i=1}^{N} \sum_{j=1}^{M} x_{ij}$, which refers to the count of satellites served. This part evaluates the use ratio of OSVs.

The second part (optimization index B_2) is written as the formula $\min m_{\mathrm{remain}}^{s}/m$, which refers to the minimum weight ($\min m_{\mathrm{remain}}^{s}$) of a single OSV's remaining fuel among all OSVs divided by the weight of the OSV (m). The purpose of the present study is to use the minimum count OSVs to serve the satellites given the count and their orbits based on the on-orbit mission. The formula m_{remain} refers to the weight of the remaining fuel after a single OSV serves satellites, and this remaining fuel can be used to keep the orbit before the OSV execute its on-orbit mission, which evaluates the ability of a single OSV keeping its orbit. If the value of $\min m_{\mathrm{remain}}^{s}$ is larger, the overall ability of all OSVs to keep orbits would be better. The maximum value of the formula $\min m_{\mathrm{remain}}^{s}/m$ is m_{fuel}/m, which the value is less than 1. In optimization index B, optimization B_1 is the main optimization index. The value of increasing or decreasing one satellite (the definition of optimization index B_1) is larger than the maximum value of optimization index B_2.

3.2. Optimization Model. The assignment optimization model of multiple OSVs serving multiple satellites is established based on the DPSO [20, 21], and the main components of the DPSO algorithm are as follows.

3.2.1. Coding of a Particle. The particles are encoded in decimal system. The length of each particle is $N \times K$. The coding of a particle is presented as Figure 4. In Figure 4, O1, O2, $\cdots$, ON are N OSVs, and Si^j refers to the j^{th} satellite served by the i^{th} OSV. The value of Si^j is obtained from the set $(0, 1, 2, \cdots, M)$ and 0 means that no satellite is served.

3.2.2. Calculation of the Fitness Function. The object function of Formula (9) is taken as the fitness function shown as

$$\max f(Ps_i) = \sum_{i=1}^{N}\sum_{j=1}^{M} x_{ij} + \frac{\min m^s_{\text{remain}}}{m}. \tag{10}$$

In Formula (10), Ps_i is the i^{th} particle.

3.2.3. Particle's Position and Velocity Update Formulas. The position of particle swarm optimization (PSO) is determined through the particle velocity, individual extreme, and global extreme, and the particle's position and velocity update formulas of the PSO algorithm are written as [22, 23]

$$\begin{cases} v_i^{h+1} = \omega \cdot v_i^h + c_1 \cdot r_1 \cdot \left(P_i^h - x_i^h\right) + c_2 \cdot r_2 \cdot \left(P_g^h - x_i^h\right), \\ x_i^{h+1} = x_i^h + v_i^{h+1}. \end{cases} \tag{11}$$

In Formula (11), P_i^h is the individual extreme in the h^{th} iteration, P_g^h is the global extreme in the h^{th} iteration, x_i^h is the fitness function value of the i^{th} particle in the h^{th} iteration, x_i^{h+1} is the fitness function value of the i^{th} particle in the $(h+1)^{\text{th}}$ iteration, v_i^h is the particle velocity value of the i^{th} particle in the h^{th} iteration, v_i^{h+1} is the particle velocity value of the i^{th} particle in the $(h+1)^{\text{th}}$ iteration, ω is the inertia weight that reflects the extent that the particle maintains the present velocity, c_1 is the cognitive coefficient that reflects the weight of the particle getting close to the individual extreme, c_2 is the social coefficient which reflects the weight the particle getting close to the global extreme, r_1 is a random number between 0 and 1, and r_2 is a random number between 0 and 1.

On the basis of the PSO, the particle position and velocity update formulas for the DPSO algorithm is written as follows:

$$Ps_i^{h+1} = c_2 \otimes F_3\left\{c_1 \otimes F_2\left[\omega \otimes F_1\left(Ps_i^h\right), P_i^h\right], P_g^h\right\}. \tag{12}$$

In Formula (12), Ps_i^h is the fitness function value of the i^{th} particle in the h^{th} iteration, and Ps_i^{h+1} is the fitness function value of the i^{th} particle in the $(h+1)^{\text{th}}$ iteration. This formula includes three formulas which are written as

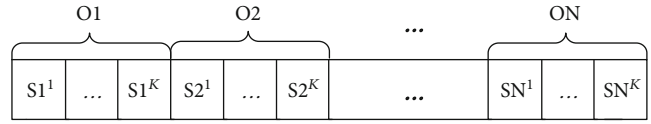

Figure 4: Coding of particle.

$$\Psi_i^h = \omega \otimes F_1\left(Ps_i^h\right) = \begin{cases} F_1\left(Ps_i^h\right), & \text{rand}\,() < \omega, \\ Ps_i^h, & \text{rand}\,() \geq \omega, \end{cases} \tag{13}$$

$$\Phi_i^h = c_1 \otimes F_2\left(\Psi_i^h, P_i^h\right) = \begin{cases} F_2\left(\Psi_i^h, P_i^h\right), & \text{rand}\,() < c_1, \\ \Psi_i^h, & \text{rand}\,() \geq c_1, \end{cases} \tag{14}$$

$$Ps_i^{h+1} = c_2 \otimes F_3\left(\Phi_i^h, P_g^h\right) = \begin{cases} F_3\left(\Phi_i^h, P_g^h\right), & \text{rand}\,() < c_2, \\ \Phi_i^h, & \text{rand}\,() \geq c_2. \end{cases} \tag{15}$$

Ψ_i^h is the value of the i^{th} particle in the h^{th} iteration after the calculation of $\omega \otimes F_1(Ps_i^h)$. The formula rand () means that a real number between 0 and 1 is randomly generated. When the condition rand () $< \omega$ is met, the formula $F_1(Ps_i^h)$ is executed which means that two integers a and b are randomly generated in interval $[0, N]$ and the values on the i^{th} particle at positions a and b are interchanged. When the condition rand () $\geq \omega$ is met, the formula Ps_i^h is executed which means that the coding values on the i^{th} particle are unchanged.

Φ_i^h is the value of the i^{th} particle in the h^{th} iteration after the calculation of $c_1 \otimes F_2(\Psi_i^h, P_i^h)$. P_i^h is the individual extreme in the t^{th} iteration. When the condition rand () $< c_1$ is met, the formula $F_2(\Psi_i^h, P_i^h)$ is executed which means that two integers a and b are randomly generated in interval $[0, N]$, and the values on Ψ_i^h at positions a and b as well as the values on P_i^h at positions a and b are interchanged. When the condition rand () $\geq c_1$ is met, the formula Ψ_i^h is executed which means that the coding values on Ψ_i^h are unchanged.

P_g^h is the global extreme in the t^{th} iteration. When the condition rand () $< c_2$ is met, the formula $F_3(\Phi_i^h, P_g^h)$ is executed which means that two integers a and b are randomly generated in interval $[0, N]$, and the values on Φ_i^h at positions a and b as well as the values on P_g^h at positions a and b are interchanged. When the condition rand () $\geq c_2$ is met, the formula Φ_i^h is executed which means that the coding values on Φ_i^h are unchanged.

The implementation steps of the DPSO algorithm are as follows.

Step 1. Set parameters: the population size M_B of the particle swarm, the count N of OSVs, the maximum count K of satellites served by a single OSV, the maximum number of

iterations T_B, the inertia weight ω, the cognitive coefficient c_1, the social coefficient c_2, and $h = 1$.

Step 2. The particle swarm population is initialized, the values on the particles are randomly generated, the fitness function $f(Ps_i)$ values of all particles are calculated, and the individual extreme P_i^h and global extreme P_g^h are recorded.

Step 3. $h = h + 1$, the position of the particles is updated through Formula (12), the fitness function $f(Ps_i)$ values of all particles are calculated, and the individual extreme P_i^h and global extreme P_g^h are updated.

Step 4. Step 3 is executed repeatedly until $(|P_g^h - P_g^{h-1}| \le 10^{-6})$ or the maximum number of iterations is reached.

4. Deployment Model

4.1. Problem Description. The deployment problem of OSVs includes two parts: one is to determine the count of OSVs, while the other is to determine the orbit elements of all OSVs.

A single OSV serves K satellites at most. If M satellites need to be served, the initial count NN of satellites can be calculated:

$$NN = \text{ceil}\left(\frac{M}{K}\right). \tag{16}$$

In Formula (16), the function ceil() means rounding up to the nearest integer. After the count of OSVs is determined by Formula (16), the orbit element orbit radius (A), orbit eccentricity ratio (e), orbit inclination angle (I), right ascension of the ascending node (Ω), argument of perigee (α), and true anomaly (f) of NN OSVs are generated according to the orbit elements of given satellites.

The optimization model to determine whether OSVs' given count and orbit elements can achieve the on-orbit mission or not is established and its optimization index is called optimization index C, which is the foundation of the deployment optimization model. If the given OSVs could not achieve the on-orbit mission, the orbit elements of all OSVs are adjusted to determine whether the on-orbit mission can be achieved and obtain the best optimization index. If the on-orbit mission could still not be achieved, the count of OSVs is increased and the orbit elements of all OSVs are adjusted again to obtain the best optimization index.

4.2. Optimization Index. Given the count and orbit elements of OSVs and taking the time interval T as the searching region for optimization variable t, the optimization model to determine whether OSVs' given count and orbit elements can achieve on-orbit mission or not is established based on the pattern search method and its optimization index is called optimization index C ($\min J_1$). In the entire searching region, optimization index B of the assignment optimization model at any time point can be calculated and minimum optimization index B is called optimization index C. The optimization index of the deployment optimization model is called optimization index D ($\max J_2$).

$$\max J_2 = \max\left(\min J_1\right) = \max\left\{\min\left[\max f(\mathbf{X})\right]\right\}. \tag{17}$$

In Formula (17), $\max f(\mathbf{X})$ is optimization index B.

The implementation steps of the pattern search [24] are as follows.

Step 1. $i = 1$. The function value $J_1(B_i)$ at base point B_i in the time interval T is calculated. The iteration step size δt and the convergence precision ε are set.

Step 2. The temporary point T_i is determined through

$$T_i = \begin{cases} B_i + \delta t, J_1(B_i + \delta t) < J_1(B_i), \\ B_i - \delta t, J_1(B_i - \delta t) < J_1(B_i) \le J_1(B_i + \delta t), \\ B_i, J_1(B_i) < \min\left[J_1(B_i + \delta t), J_1(B_i - \delta t)\right]. \end{cases} \tag{18}$$

Formula (18) means that the temporary point T_i is determined in point $(B_i - \delta t)$, point $(B_i + \delta t)$, and point B_i which has a minimum value of J_1. The temporary point T_i is defined as base point B_{i+1}. i^{th} is the base point B_i pointing to base point B_{i+1}.

Step 3. If the condition $J_1(T_i) < J_1(B_i)$ is met, the $(i+1)^{\text{th}}$ vector is twice longer than the i^{th} vector. $i = i + 1$. The temporary point T_i is written as

$$T_i = B_{i-1} + 2(B_i - B_{i-1}). \tag{19}$$

The temporary point T_i is defined as the base point B_{i+1} which is taken as the starting point of the i^{th} vector. If the condition $J_1(T_i) \ge J_1(B_i)$ is met, the $(i+1)^{\text{th}}$ vector is half of the i^{th} vector and its starting point becomes the base point B_{i-1}.

Step 4. Steps 2 and 3 are repeatedly executed. If the value of T_i remains unchanged, the iteration step size δt will be half until it reaches a certain degree of accuracy. The algorithm ends.

4.3. Optimization Model. Given the count of OSVs, the optimized orbit elements are obtained by adjusting orbit elements of all OSVs. The deployment problem of OSVs is a high-dimension nonlinear problem which nests the optimization problems in Sections 2.2, 3.2, and 4.2.

If the inertia weight ω is bigger, the global search capability of the PSO algorithm will be stronger. If the inertia weight ω is smaller, the local search capability of the PSO algorithm will be stronger. Considering that the searching space of deployment problem of OSVs is large and the optimization variables are sensitive to the initial values, a larger inertial weight ω will be chosen to enhance the global search capability. Considering that the fuel taken by the OSV is expensive, enhancing the local search capability to obtain the better local results will also be very important. Hence, the fuzzy control system is introduced to adjust the inertia weight ω, which can balance the global search capability and local search capability.

Table 1: The fuzzy control rule base.

Number	$\overline{\text{fit}}$	ω	$\Delta\omega$
1	Small	Small	Medium
2	Small	Medium	Small
3	Small	Large	Small
4	Medium	Small	Large
5	Medium	Medium	Medium
6	Medium	Large	Small
7	Large	Small	Large
8	Large	Medium	Medium
9	Large	Large	Small

Table 2: Initial orbit elements of satellites.

Satellite	A (km)	e	I (°)	α (°)	Ω (°)	f (°)
S1	9306	0.238550	28.321	224.138	307.927	135.862
S2	7059	0.000225	98.084	76.815	185.081	283.332
S3	9237	0.000165	52.006	171.386	350.005	188.701
S4	7828	0.001219	101.618	87.562	293.441	282.681
S5	7707	0.054088	99.031	56.562	347.938	308.882
S6	7554	0.002882	90.045	245.844	85.215	113.968
S7	8428	0.201645	82.982	262.555	29.901	44.341
S8	9501	0.213200	28.226	34.607	152.421	325.392
S9	9401	0.016556	64.822	302.192	337.387	56.304
S10	9939	0.321639	56.917	231.629	218.498	128.371
S11	10687	0.355917	57.022	347.224	317.457	122.776
S12	6874	0.000555	97.420	104.012	316.284	204.781
S13	7125	0.003924	42.084	51.815	255.081	30.200
S14	8838	0.000249	98.178	69.604	155.872	150.009
S15	9123	0.013793	52.674	109.398	23.270	200.020
S16	7685	0.004693	37.477	52.707	158.017	60.000
S17	9215	0.489300	55.371	128.871	305.501	80.000
S18	7453	0.004218	23.618	77.562	293.441	50.101
S19	9675	0.054089	99.031	56.562	187.938	160.032
S20	7534	0.003002	91.145	239.735	84.325	113.000
S21	6903	0.000549	96.390	110.033	325.324	203.070

Table 3: Initial orbit elements of OSVs.

OSV	A (km)	e	I (°)	α (°)	Ω (°)	f (°)
O1	7751	0.000120	32.223	56.232	76.223	255.520
O2	8400	0.000225	78.002	102.332	253.010	122.243
O3	8526	0.004120	76.924	154.257	73.020	243.000
O4	9201	0.370056	55.223	180.922	223.001	95.620
O5	7002	0.004021	87.235	102.556	330.765	88.598
O6	8002	0.529030	145.667	156.007	200.321	270.321
O7	9923	0.052301	120.023	254.124	122.232	55.220
O8	9001	0.006322	78.336	155.336	95.663	45.692

Table 4: Parameters of the fuzzy controller.

Parameter	Fuzzy language	a	b
$\overline{\text{fit}}$	Small	0	0.06
	Medium	0.05	0.4
	Large	0.3	1
ω	Small	0.2	0.6
	Medium	0.4	0.9
	Large	0.6	1.1
$\Delta\omega$	Small	-0.12	-0.02
	Medium	-0.04	0.04
	Large	0.00	0.05

The essence of the FAPSO algorithm is updating the inertia weight ω through the fuzzy controller. The updated inertial weight ω_{updated} is determined through the fuzzy controller in three steps.

$$\overline{\text{fit}} = \frac{\text{fit} - \text{fit}_{\min}}{\text{fit}_{\max} - \text{fit}_{\min}}. \tag{20}$$

Step 1. The input variables of the fuzzy controller are translated into fuzzy variables. The two input variables are the inertial weight ω and the normalized fitness function values $\overline{\text{fit}}$ of particles in the present iteration.

In Formula (20), fit is the fitness function value of one particle in the present iteration, $\text{fit}_{\min}$ is the minimum fitness function value in the present iteration, and $\text{fit}_{\max}$ is the maximum fitness function value in the present iteration.

The fuzzy variables of these two input variables are calculated through the membership functions [25, 26].

$$f_s = \begin{cases} 1, & x < a_S, \\ \dfrac{b_S - x}{b_S - a_S}, & a_S \le x \le b_S, \\ 0, & x > b_S, \end{cases} \tag{21}$$

$$f_M = \begin{cases} 0, & x < a_M, \\ 2\dfrac{x - a_M}{b_M - a_M}, & a_M \le x \le \dfrac{a_M + b_M}{2}, \\ 2\dfrac{b_M - x}{b_M - a_M}, & \dfrac{a_M + b_M}{2} \le x \le b_M, \\ 0, & x > b_M, \end{cases} \tag{22}$$

$$f_L = \begin{cases} 0, & x < a_L, \\ \dfrac{x - a_L}{b_L - a_L}, & a_L \le x \le b_L, \\ 1, & x > b_L. \end{cases} \tag{23}$$

Formulas (21), (22), and (23) are the membership functions that correspond to fuzzy language "small," "medium," and "large," respectively. a_S and b_S are the constants when

Table 5: Orbit elements of OSVs after simulations.

OSV	A (km)	e	I (°)	α (°)	Ω (°)	f (°)
O1	7509	0.000221	28.100	55.221	125.122	305.600
O2	8144	0.000198	82.998	115.419	320.385	65.768
O3	7792	0.000221	100.325	112.366	102.981	200.421
O4	8879	0.250022	55.348	200.000	250.887	120.133
O5	6998	0.005411	96.951	156.052	349.870	149.551
O6	7331	0.452239	38.002	200.333	250.554	35.010
O7	10049	0.240000	53.239	300.256	320.457	225.000
O8	9514	0.151000	97.000	265.107	145.932	102.000

Table 6: Results of deployment.

Time (h)	OSV	Satellite	J_2
9336.450	O1	S8, S1, S18	21.106
	O2	S7, S20, S6	
	O3	S2, S4	
	O4	S10, S17, S15	
	O5	S12, S21, S5	
	O6	S13, S16	
	O7	S11, S9, S3	
	O8	S19, S14	

two input variables ω or $\overline{\text{fit}}$ are within the range of fuzzy language "small." a_M and b_M are the constants when two input variables ω or $\overline{\text{fit}}$ are within the range of fuzzy language "medium." a_L and b_L are the constants when two input variables ω or $\overline{\text{fit}}$ are within the range of fuzzy language "large."

Step 2. The values of two input variables ω and $\overline{\text{fit}}$ and their fuzzy variables are used to calculate the fuzzy control variable $\Delta\omega$ according to the fuzzy reasoning algorithm and the fuzzy control rule base.

The fuzzy reasoning algorithm can be expressed through

$$\mu_{R(X,Y)}(x, y) = \mu_A(x) \wedge \mu_B(y). \tag{24}$$

In Formula (24), x is the input variable ω, y is the input variable $\overline{\text{fit}}$, X is the set where x is in, Y is the set where y is in, $R(X, Y)$ is the implication relationship between X and Y, $\mu_A(x)$ is the membership function value of x, $\mu_B(y)$ is the membership function value of y, and "$\wedge$" is the operational symbol for choosing the smaller value between $\mu_A(x)$ and $\mu_B(y)$.

The fuzzy control rule base is described in Table 1.

$$v_{\text{out}} = \frac{\int_V v\mu_V(v)dv}{\int_V \mu_V(v)dv}. \tag{25}$$

Step 3. The fuzzy control variable $\Delta\omega$ is translated into the precise value after the defuzzification procedure. The precise value v_{out} of a particle is the centre point's x-coordinate value of the area of the curve $f(\Delta\omega) = \mu_{R(X,Y)}(x, y)$ in Step 2 and the x-coordinate.

In Formula (25), $\mu_V(v)$ is $\mu_{R(X,Y)}(x, y)$, V is the area of the curve $f(\Delta\omega) = \mu_{R(X,Y)}(x, y)$ in Step 2 and the x-coordinate, and v is the x-coordinate value. The updated inertial weight ω_{updated} is written as

$$\omega_{\text{updated}} = \omega + v_{\text{out}}. \tag{26}$$

The implementation steps of the FAPSO algorithm are as follows.

Step 1. Set parameters: the population size M_C of the particle swarm, the count N of OSVs, the maximum number of iterations T_C, the inertia weight ω, the cognitive coefficient c_1, the social coefficient c_2, and $h = 1$. The coding of the i^{th} particle is $Ps_i = [x_i^{11}, \cdots, x_i^{16}, \cdots, x_i^{j1}, \cdots, x_i^{j6}, \cdots, x_i^{N6}, \cdots, x_i^{N6}]$, the expression x_i^{j6} refers to the k^{th} orbit element of the j^{th} OSV in the i^{th} particle, and $k = 1, 2, 3, 4, 5, 6$ corresponds to the six orbit elements A, e, I, Ω, α, and f, respectively.

Step 2. The particle swarm population is initialized, the values on the particles are randomly generated, the fitness function values of all particles are calculated using Formula (17), and the individual extreme P_i^h and the global extreme P_g^h are recorded.

Step 3. The inertial weight ω and the normalized fitness function values $\overline{\text{fit}}$ in the h^{th} iteration are calculated and the updated inertial weight ω_{updated} is determined through the fuzzy controller in three steps. $h = h + 1$. The updated inertial weight ω_{updated} is placed into Formula (11) to calculate the fitness function value of the particle in the h^{th} iteration. The fitness function values of all particles are calculated, and the individual extreme P_i^h and global extreme P_g^h are recorded.

Step 4. Step 3 is repeatedly executed until $(|P_g^h - P_g^{h-1}| \le 10^{-6})$ or the maximum number of iterations is reached.

Step 5. If the optimization result cannot achieve the on-orbit mission, one OSV is added to the present OSVs and Steps 1–4 are executed. If the optimization result can achieve the on-orbit mission, the cycle calculation is ended.

5. Simulations and Analyses of Results

5.1. Simulations. The initial orbit elements of all satellites are presented in Table 2. The search range orbit elements (A, e, I, Ω, α, and f) of the OSVs determined according to the orbit elements of all satellites are [6700 km,11000 km], [0, 1), [0°, 180°], [0°, 360°], [0°, 360°], and [0°, 360°], respectively. The initial orbit elements of OSVs are presented in Table 3. The weight of a single OSV is 1000 kg. A single OSV serves three satellites at most and carries 600 kg of fuel. The specific

Table 7: OSVs' manoeuvre time, rendezvous time, and the weight of remaining fuel.

OSV	t_1 (s)	t_2 (s)	t_3 (s)	t_4 (s)	t_5 (s)	t_6 (s)	m^s_{remain} (kg)
O1	51692.558	67975.303	104345.873	109193.762	193120.182	198267.663	199.102
O2	175202.944	187625.628	219612.145	225239.147	250263.743	255990.129	111.257
O3	17001.794	148722.117	218123.229	239686.355	—	—	105.759
O4	15887.24	26912.829	31971.051	37423.464	47426.589	59549.761	126.350
O5	3605.5	39687.429	51749.981	63848.707	—	—	150.445
O6	33806.931	60783.397	97927.029	107616.136	179462.553	216913.012	232.828
O7	22523.519	44028.819	105397.328	126423.851	151691.717	186117.106	107.141
O8	31522.046	72061.837	75662.262	130570.746	—	—	164.904

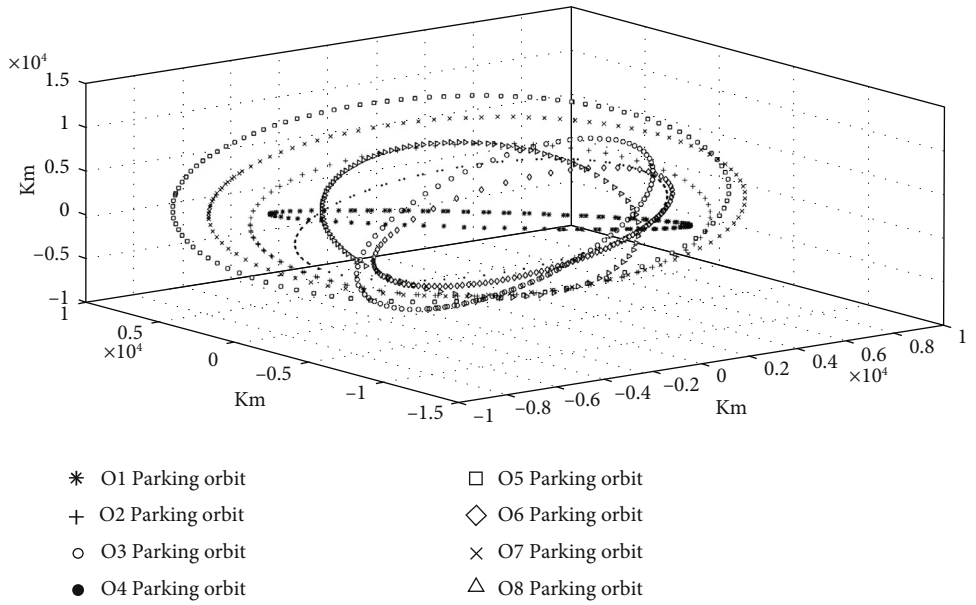

Figure 5: Parking orbits of all OSVs.

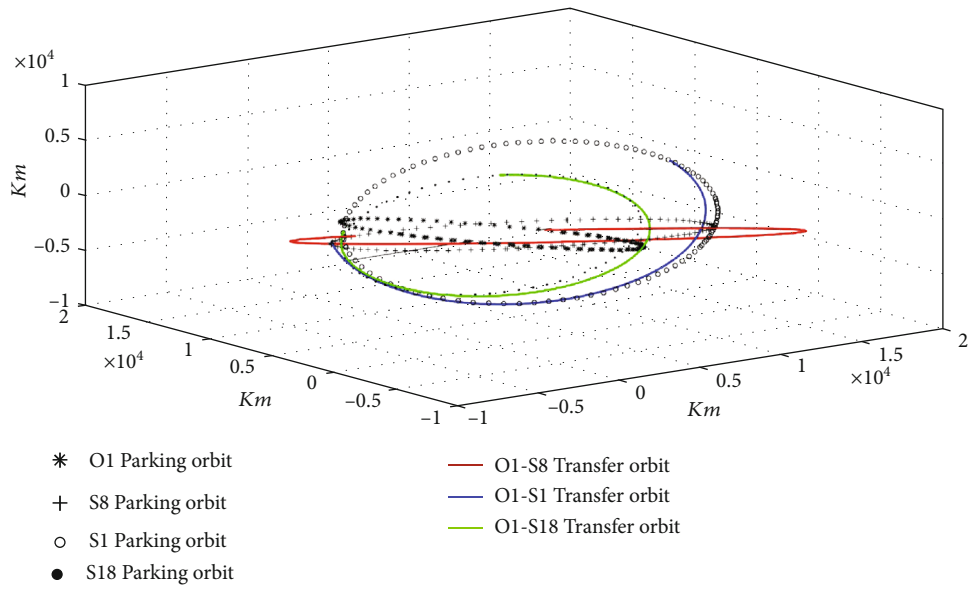

Figure 6: Transfer orbit of O1 serving S8, S1, and S18.

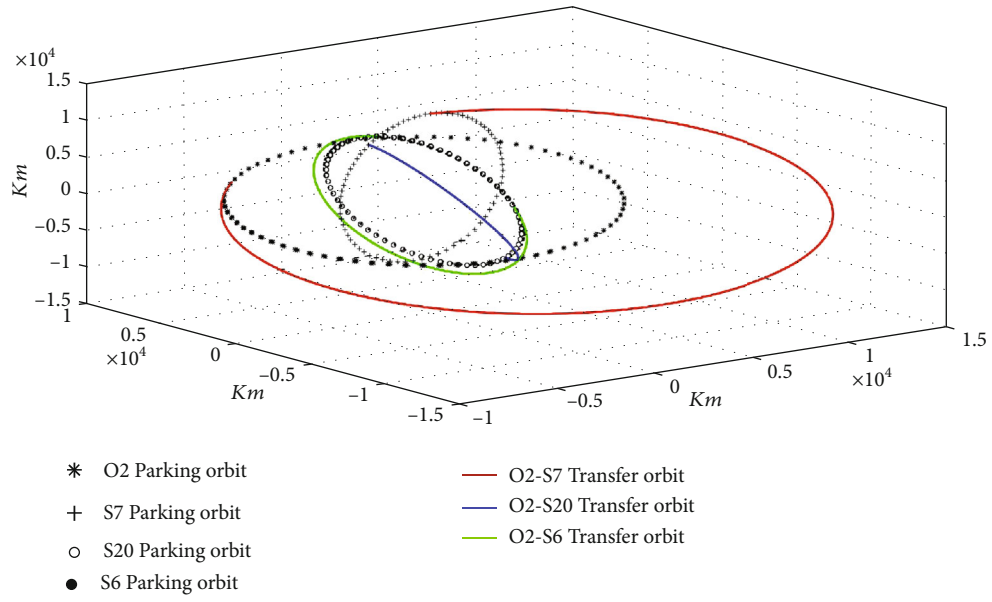

Figure 7: Transfer orbit of O2 serving S7, S20, and S6.

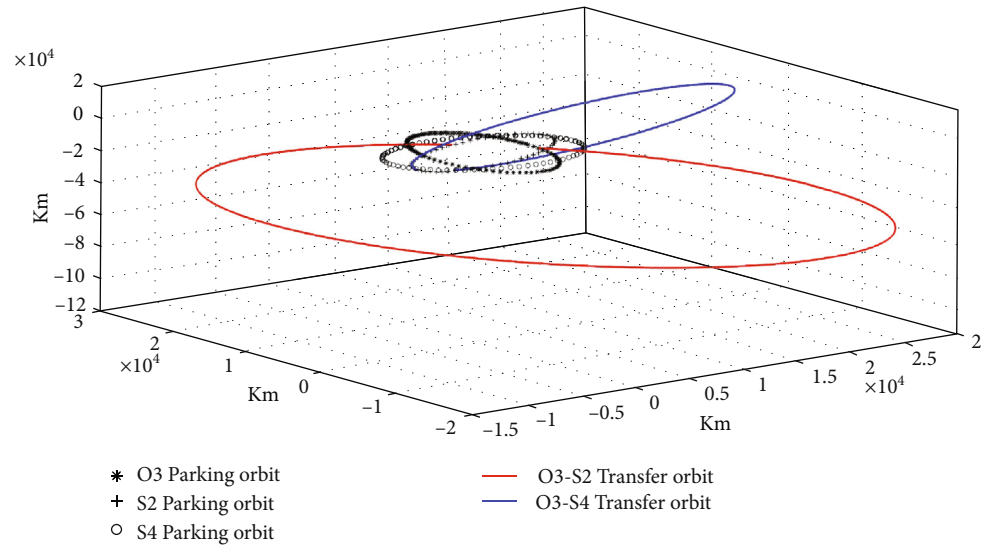

Figure 8: Transfer orbit of O3 serving S10, S17, and S15.

impulse is 3000 seconds. The duration of time to serve a satellite is more than an hour. The on-orbit mission should be achieved within 72 hours.

In the simulation of the transfer orbit optimization model, the parameters of GA are taken from literature written by Tesarova & Vokalova [12]. The population size M_A is 20, the weight of the fuel used to keep the orbit m_{keep} is 50 kg, the maximum number of iterations T_A is 50, the crossover probability P_c is 0.85, and the mutation probability P_m is 0.1.

In the simulation of assignment optimization model, the parameters of DPSO are taken from literature written by Pradeepmon et al. [11]. The population size M_B is 10, the maximum number of iterations T_B is 50, the inertia weight ω is 0.75, the cognitive coefficient c_1 is 2, and the social coefficient c_2 is 2.

In the simulation of deployment model, the parameters of PSO are taken from literature written by Singh et al. [13]. The population size M_C is 10, the maximum number of iterations T_C is 50, the inertia weight ω is 0.75, the cognitive coefficient c_1 is 2, and the social coefficient c_2 is 2. The particle velocities corresponding to the six orbit elements A, e, I, Ω, α, and f are 50000, 0.00001, 10, 10, 10, and 10, respectively. In the optimization model to determine whether OSVs' given count and orbit elements can achieve on-orbit mission or not, the iteration step size δt of the pattern search is $(24 \times 60 \times 60)$ seconds and the time interval T is

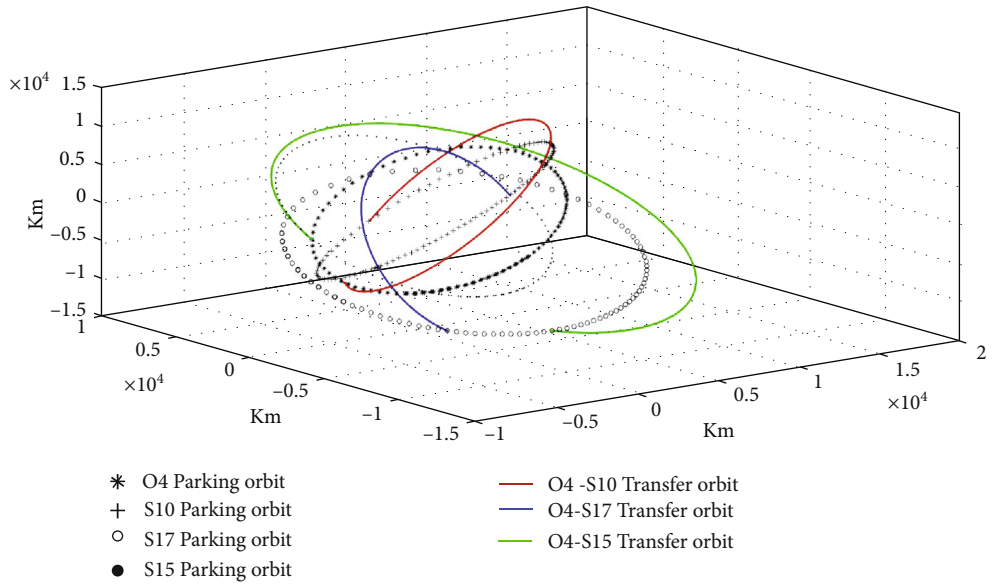

Figure 9: Transfer orbit of O4 serving S12, S21, and S5.

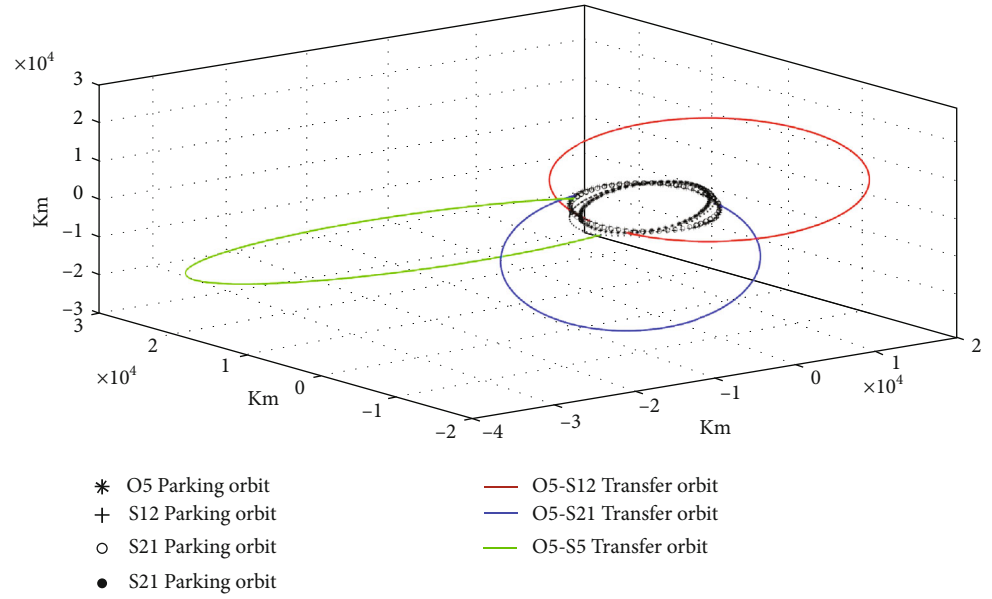

Figure 10: Transfer orbit of O5 serving S11, S9, and S3.

$(2 \times 365 \times 24 \times 60 \times 60)$ seconds. The parameters of the fuzzy controller are shown in Table 4.

We design the comparison experiments between the proposed FAPSO algorithm and GA, PSO, PSO+TS, and GA+SA algorithms. The parameters of these algorithms are successively taken from the existing literatures mentioned in the present paper [12–15]. 30 independent runs are performed for each algorithm.

5.2. Analyses of Results. The results of the 10^{th} experiment presented in Tables 5–7 and Figures 5–13 are obtained using the FAPSO algorithm. The obtained orbit elements of OSVs are presented in Table 5, and the parking orbits of OSVs are presented in Figure 5. The deployment results of OSVs are presented in Table 6 including the value of optimization D (max J_2), the time to obtain max J_2, and the assignment results of OSVs. On the basis of results presented in Table 6, OSVs' manoeuvre time, rendezvous time, and the weight of remaining fuel are presented in Table 7. The transfer orbits of OSVs are presented in Figures 6–13. The convergence curves (the average values of 30 independent experiments) of the FAPSO algorithm, GA, PSO algorithm, PSO+TS algorithm, and GA+SA algorithm are presented in Figure 14.

In Table 6, the time to obtain optimization index D (max J_2) is after 9336.450 hours. O1, O2, O4, O6, and O7

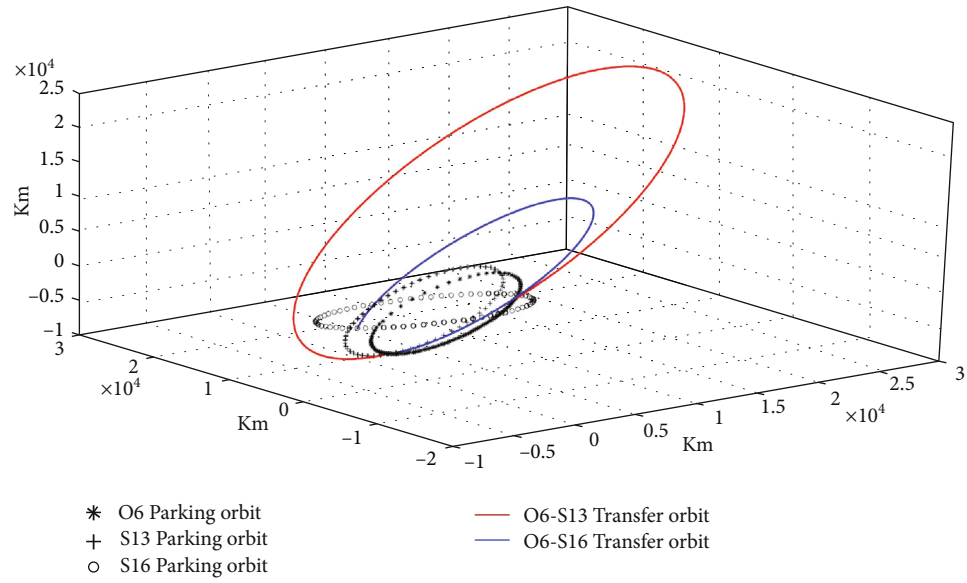

Figure 11: Transfer orbits of O6 serving S13 and S16.

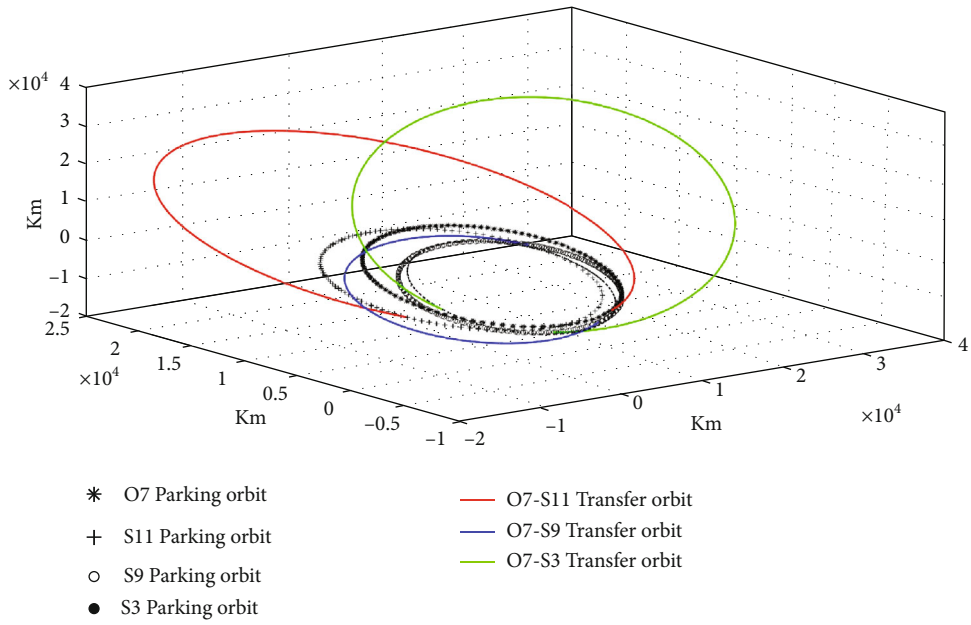

Figure 12: Transfer orbit of O7 serving S19 and S14.

each serve 3 satellites. O3, O5, and O8 each serve 2 satellites. O1 serves S8, S1, and S18 in sequence. O2 serves S7, S20, and S6 in sequence. O3 serves S2 and S4 in sequence. O4 serves S10, S17, and S15 in sequence. O5 serves S12, S21, and S5 in sequence. O6 serves S13 and S16 in sequence. O7 serves S11, S9, and S3 in sequence. O8 serves S19 and S14 in sequence. In Table 7, t_1 and t_2 are the manoeuvre time and rendezvous time to serve the 1st satellite, respectively, t_3 and t_4 are the manoeuvre time and rendezvous time to serve the 2nd satellite, respectively, t_5 and t_6 are the manoeuvre time and rendezvous time to serve the 3rd satellite, respectively, and m^s_{remain} is the weight of remaining fuel after the s^{th} OSV serving the satellites. In Figure 14, the convergence results of the FAPSO algorithm, PSO algorithm, PSO+TS algorithm, GA, and GA+SA algorithm are 21.106, 21.083, 21.084, 21.080, and 21.094, respectively. After the convergence results are entered into the objection function of Formula (9), the weight of the remaining fuel obtained through the FAPSO algorithm is 23 kg, 22 kg, 26 kg, and 12 kg more than that obtained by the PSO algorithm, PSO+TS algorithm, GA, and GA+SA algorithm, respectively.

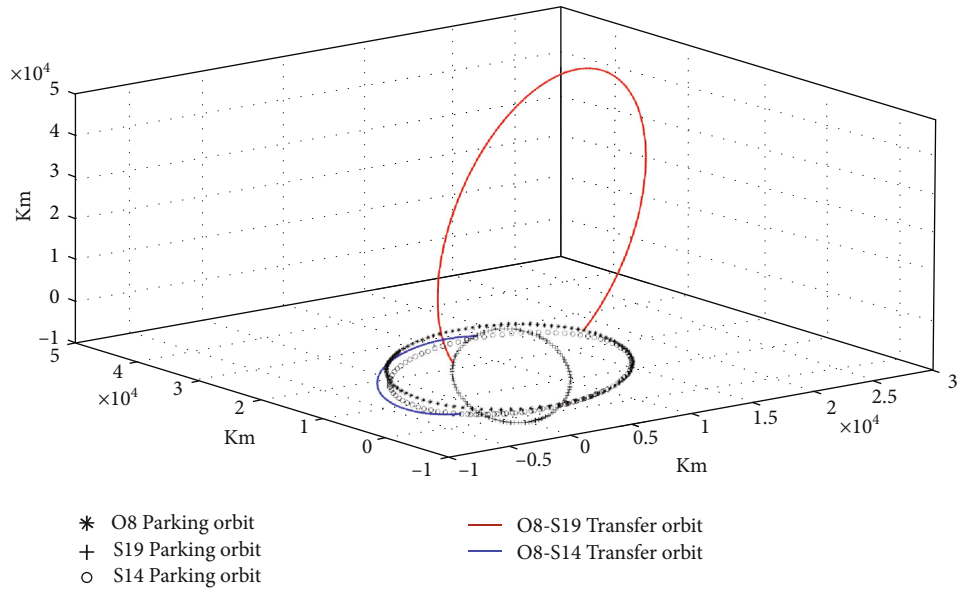

Figure 13: Transfer orbit of O8 serving S2 and S4.

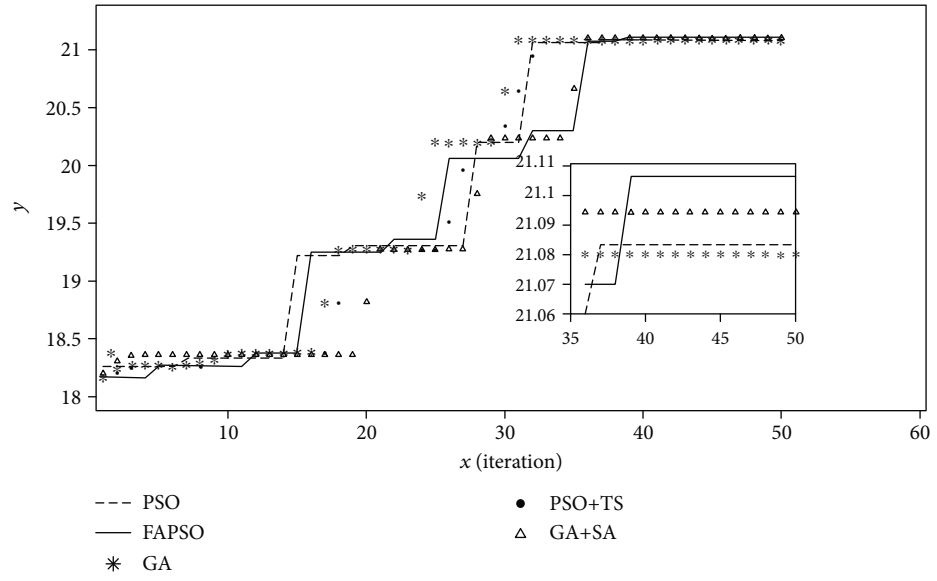

Figure 14: The convergence curves of the FAPSO algorithm and other algorithms.

6. Conclusion and Further Research

The deployment of multiple OSVs is a complex multiple nested optimization problem, which is systematically solved by a series of intelligent algorithms in the present study. The deployment model includes the transfer orbit optimization model and the assignment optimization model. The transfer orbit optimization is a continuous optimization problem, which is solved by the intelligence algorithm GA. The assignment optimization is a discrete optimization problem, which is solved by the intelligence algorithm the DPSO algorithm. The deployment is a continuous optimization problem, which is solved by the FAPSO algorithm. The FAPSO algorithm, which has a better convergence result than that obtained using the other similar optimization algorithms, can effectively solve the deployment problem of OSVs.

In further studies, some aspects should be considered such as the finite thrust engine of the OSV, the influence on the parking orbits because of the disturbing force, and optimization algorithms getting the simulation results faster.

Notations

t: Uncertain time
T: Time interval
$t_1, t_3, \cdots, t_{2n-1}$: Manoeuvre times
$t_2, t_4, \cdots, t_{2n}$: Rendezvous times
$T_{\max}$: Maximum time interval for finishing the on-orbit mission
t_{serve}: Time interval for the OSV serving one satellite
$\mathbf{r}_1$: Position vector at point P_1
$\mathbf{r}_2$: Position vector at point P_2
$\mathbf{v}_0$: Velocity of the original orbital at point P_1
$\mathbf{v}_1$: Velocity vector at point P_1
$\mathbf{v}_2$: Velocity vector, respectively, at point P_2
$\mathbf{v}_3$: Velocity of the target orbit at point P_2
Δv_{i1}: The 1st velocity increment of the OSV serving the i^{th} satellite at the manoeuvre time
Δv_{i2}: The 2nd velocity increment of the OSV serving the i^{th} satellite at the rendezvous time
m: Weight of a single OSV
m_{fuel}: Weight of the fuel carried by the OSV
m_{else}: Weight of the other parts
m_{remain}: Weight of the remaining fuel after the on-orbit mission is achieved
Δm_i: Weight of the consumed fuel
m_{keep}: Weight of fuel used to keep the orbit
m_{i-1}: Weight of the remaining fuel after a single OSV serves $i-1$ satellites
m^s_{remain}: Weight of the remaining fuel after the s^{th} OSV serves the satellites
n: Count of the transfer orbit segments
k: Count of satellites served
K: Count of satellites that single OSV can serve
Ps_i: i^{th} particle
Ps_i^h: Fitness function value of the i^{th} particle in the h^{th} iteration
Ps_i^{h+1}: Fitness function value of the i^{th} particle in the $(h+1)^{\text{th}}$ iteration
P_i^h: Individual extreme in the t^{th} iteration
P_g^h: Global extreme in the t^{th} iteration
$\max f(\mathbf{X})$: Optimization index B
fit: Fitness function value of one particle in the present iteration
$\text{fit}_{\min}$: Minimum fitness function value in the present iteration
$\text{fit}_{\max}$: Maximum fitness function value in the present iteration
Ψ_i^h: Value of the i^{th} particle in the h^{th} iteration after the calculation of $\omega \otimes F_1(Ps_i^h)$
Φ_i^h: Value of the i^{th} particle in the h^{th} iteration after the calculation of $c_1 \otimes F_2(\Psi_i^h, P_i^h)$
ω: Inertia weight
x: Input variable ω
y: Input variable $\overline{\text{fit}}$
X: Set where x is in
Y: Set where y is in
$\mu_A(x)$: Membership function value of x
$\mu_B(y)$: Membership function value of y
$\wedge$: Operational symbol for choosing the smaller value between $\mu_A(x)$ and $\mu_B(y)$.

Data Availability

The simulation data used to support the findings of this study are included within the article.

Conflicts of Interest

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

Acknowledgments

This article is supported by the special research fund of Xi'an High-Tech Institute and cosupported by the National Natural Science Foundation of China (61603398).

References

[1] X. Q. Chen, J. P. Yuan, W. Yao, and Y. Zhao, *On-Orbit Service for Spacecraft*, China Astronautic Publishing House, Beijing, China, 2009.

[2] F. Sellmaier, T. Boge, J. Spurmann, S. Gully, T. Rupp, and F. Huber, "On-orbit servicing missions: challenges and solutions for spacecraft operations," in *In Proceedings of the AIAA space operations 2010 conference, space operations: exploration, scientific utilization, and technology development*, pp. 213–233, 2010.

[3] D. Walt, *On-Orbit Servicing of Space Systems*, Krieger Publishing Company, Florida, USA, 1993.

[4] Y. Li, J. B. Yao, C. J. Xin, and X. C. Su, "The approaching process analysis of on-orbit service vehicle to circular co-planarity orbit target," *Aerospace Control*, vol. 33, no. 4, pp. 56–61, 2015.

[5] J. A. Andrew, B. S. David, and J. H. Terry, "Early mission design of transfers to halo orbits via particle swarm optimization," *Journal of the Astronautical Sciences*, vol. 63, no. 2, pp. 81–102, 2016.

[6] M. C. F. Bazzocchi and M. R. Emami, "Stochastic optimization of asteroid three-dimensional trajectory transfer," *Acta Astronautica*, vol. 152, pp. 705–718, 2018.

[7] M. J. Grant and R. D. Braun, "Rapid indirect trajectory optimization for conceptual design of hypersonic missions," *Journal of Spacecraft and Rockets*, vol. 52, no. 1, pp. 177–182, 2015.

[8] B. Andrea and C. Christian, "A hybrid, self-adjusting search algorithm for optimal space trajectory design," *Advances in Space Research*, vol. 50, no. 4, pp. 471–488, 2012.

[9] K. Daneshjou, A. A. Mohammadi-dehabadi, and M. Bakhtiari, "Mission planning for on-orbit servicing through multiple servicing satellites: a new approach," *Advances in Space Research*, vol. 60, no. 6, pp. 1148–1162, 2017.

[10] Y. Zhang and Q. Zhang, "On-orbit servicing task allocation for multi-spacecrafts using HDPSO," *Applied Mechanics and Materials*, vol. 538, pp. 150–153, 2014.

[11] T. G. Pradeepmon, R. Sridharan, and V. V. Panicker, "Development of modified discrete particle swarm optimization algorithm for quadratic assignment problems," *International Journal of Industrial Engineering Computations*, vol. 9, no. 4, pp. 491–508, 2018.

[12] B. Tesarova and A. Vokalova, "Using a genetic algorithm in a process of optimizing the deployment of radio stations," *Future Data and Security Engineering*, vol. 106, no. 46, pp. 110–118, 2017.

[13] P. Singh, S. Mini, and K. Sabale, "Particle swarm optimization for the deployment of directional sensors," *Swarm, Evolutionary, and Memetic Computing*, vol. 98, no. 73, pp. 167–175, 2016.

[14] A. Ahmadian, A. Elkamel, and A. Mazouz, "An improved hybrid particle swarm optimization and tabu search algorithm for expansion planning of large dimension electric distribution network," *Energies*, vol. 12, no. 16, p. 3052, 2019.

[15] J. Zhang, X. H. Zhang, and H. X. Wu, "Genetic simulated annealing algorithm for optimal deployment of flow monitors," in *In Proceedings of the third international conference on natural computation*, pp. 398–405, 2007.

[16] P. W. J. Schumacher, C. Sabol, C. C. Higginson, and K. T. Alfriend, "Uncertain Lambert problem," *Journal of Guidance, Control, and Dynamics*, vol. 38, no. 9, pp. 1573–1584, 2015.

[17] V. A. Kostenko and A. V. Frolov, "Self-learning genetic algorithm," *Journal of Computer and Systems Sciences International*, vol. 54, no. 4, pp. 525–539, 2015.

[18] B. D. Liu, "Genetic algorithms," *Theory and Practice of Uncertain Programming*, vol. 239, pp. 9–17, 2009.

[19] E. Mehmed, "A new approach for the genetic algorithm," *Journal of Statistical Computation and Simulation*, vol. 79, no. 3, pp. 275–297, 2009.

[20] R. Karthi, S. Arumugam, and K. R. Kumar, "Discrete particle swarm optimization algorithm for data clustering," *Nature Inspired Cooperative Strategies for Optimization*, vol. 236, 2009.

[21] A. Saman and S. Rifat, "Discrete particle swarm optimization method for the large-scale discrete time–cost trade-off problem," *Expert Systems with Applications*, vol. 51, pp. 177–185, 2016.

[22] A. Ismail and A. P. Engelbrecht, "Self-adaptive particle swarm optimization," *Simulated Evolution and Learning*, vol. 7673, pp. 228–237, 2012.

[23] S. Kiranyaz, T. Ince, and M. Gabbouj, "Particle swarm optimization," *Multidimensional Particle Swarm Optimization for Machine Learning and Pattern Recognition*, vol. 15, pp. 45–82, 2014.

[24] R. K. Sahu, S. Panda, and S. Padhan, "A novel hybrid gravitational search and pattern search algorithm for load frequency control of nonlinear power system," *Applied Soft Computing*, vol. 29, pp. 310–327, 2015.

[25] E. Kayacan, A. Sarabakha, S. Coupland, R. John, and M. A. Khanesar, "Type-2 fuzzy elliptic membership functions for modeling uncertainty," *Engineering Applications of Artificial Intelligence*, vol. 70, pp. 170–183, 2018.

[26] T. Mitsuishi and K. Saigusa, "Defuzzification of periodic membership function on circular coordinates," *International Journal of Mathematical, Computational, Natural and Physical Engineering*, vol. 8, no. 10, pp. 1246–1249, 2014.