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

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 directionals. 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 \( \Delta t \) seconds to serve the satellite, \( r_1 \) and \( v_1 \) are its position vector and velocity vector, respectively, at point \( P_1 \), and \( r_2 \) and \( v_2 \) are its position vector and velocity vector, respectively, at point \( P_2 \). The determination of the 1st velocity increment \( \Delta v_{11} \) (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 \( \Delta v_{12} \) 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.

\[
\Delta v_{11} = v_1 - v_0, \quad (1)
\]
\[
\Delta v_{12} = v_3 - v_2. \quad (2)
\]

In Formula (1) and Formula (2), \( v_0 \) is the velocity of the OSV at point \( P_1 \) on the original orbit, \( v_1 \) is the velocity of the OSV at point \( P_1 \) on the transfer orbit after pushing by the 1st pulse thrust, \( v_2 \) is the velocity of the OSV at point \( P_2 \) on the transfer orbit, and \( v_3 \) 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.

\[
m = m_{\text{fuel}} + m_{\text{else}}. \quad (3)
\]

In Formula (3), \( m \) is the weight of a single OSV, \( m_{\text{fuel}} \) is the weight of the fuel carried by the OSV, and \( m_{\text{else}} \) is the weight of the other parts.
Given the sequence of serving satellites, the mathematical model can be written as

\[ \text{max } J = m_{\text{remain}} \]

\[ t_1 < t_2 < t_3 < t_4 < \cdots < t_{2n-1} < t_{2n} < T_{\text{max}} - t_{\text{serve}} \]

\[ t_3 - t_2 \geq t_{\text{serve}}, \cdots, t_{2n-1} - t_{2n-2} \geq t_{\text{serve}} \]

\[ \sum_{i=1}^{n} \Delta m_i \leq m_{\text{fuel}} - m_{\text{keep}}. \]

(4)

In Formula (4), \( J \) is the optimization index A, \( m_{\text{remain}} \) 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, \( T_{\text{max}} \) is the maximum time interval for finishing the on-orbit mission, \( t_{\text{serve}} \) 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), \( \Delta m_i \) is the weight of the consumed fuel in the \( i^{th} \) segment, and \( m_{\text{keep}} \) is the weight of fuel used to keep the orbit. \( m_{\text{remain}} \) can be written as Formula (5) and \( \Delta m_i \) can be written as Formula (6).

\[ m_{\text{remain}} = m_{\text{fuel}} - \sum_{i=1}^{k} \Delta m_i, k \leq K, \]

(5)
\[ \Delta m_i = \left( m_{i-1} \left[ 1 - \exp \left( -\frac{\Delta v_{i1}}{g_i t_{sp}} \right) \right] \right) \left[ 1 - \exp \left( -\frac{\Delta v_{i2}}{g_i t_{sp}} \right) \right], \quad i \leq K. \]  

(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, \( t_{sp} \) is the specific impulse of the OSV, \( g_i \) is the acceleration of the earth, \( m_i \) is the weight of the remaining fuel after the OSV serves the \( i \)th satellite, \( \Delta v_{i1} \) is the 1st velocity increment of the OSV serving the \( i \)th satellite at the manoeuvre time, and \( \Delta v_{i2} \) is the 2nd 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 \( m_{\text{remain}} \)) 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 \( t_{10}, t_{21}, \ldots, t_{2n(2n-1)} \) are written as

\[
\begin{align*}
  t_{2i(2i-1)} &= t_{2i} - t_{2i-1} \geq 3600, \quad i = 1, 2, \ldots, n, \\
  t_{2i-1(2i-2)} &= t_{2i-1} - t_{2i-2} \geq 0, \\
  i &= 1, 2, \ldots, n.
\end{align*}
\]

(7)

In Formula (7), \( n \) is the count of the transfer orbit segments. Furthermore, \( t_{10}, t_{21}, \ldots, t_{2n(2n-1)} \) are taken as genes encoded to a chromosome (Figure 3). The initial populations are randomly generated. The population size \( M_n \), 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 (\( |f(R_i) - f(R_{i-1})| \leq 10^{-6} \)) 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

Assumption 1. There are \( N \) OSVs that are remarked as \( O = \{O1, O2, \ldots, ON\} \), there are \( M \) satellites that are remarked as \( S = \{S1, S2, \ldots, SM\} \), a single OSV serves \( K \) satellites at most, and a single satellite can be served once at most. The assignment matrix \( X \) is written as

\[
x_{ij} = \begin{cases} 
1, & \text{if serving } S_j, \\
0, & \text{if not serving } S_j.
\end{cases}
\]

(8)

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

\[
\max f(X) = \sum_{i=1}^{N} \sum_{j=1}^{M} x_{ij} + \min m_{\text{remain}}^i \over m
\]

(9)

\[
\sum_{j=1}^{M} x_{ij} \leq K, \forall i = 1, 2, \ldots, N
\]

s.t.

\[
\sum_{i=1}^{N} x_{ij} \leq 1, \forall j = 1, 2, \ldots, M
\]

\[
m_{\text{remain}}^i \geq m_{\text{keep}}, \forall s = 1, 2, \ldots, N.
\]

In Formula (9), \( \max f(X) \) 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, \( m_{\text{remain}} \) is the weight of the remaining fuel after the \( s \)th OSV serves the satellites, and \( m_{\text{keep}} \) 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 \( \min m_{\text{remain}}^i / m \), which refers to the minimum weight \( \min m_{\text{remain}} \) 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 \( m_{\text{remain}} \) 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 \( \min m_{\text{remain}} \) is larger, the overall ability of all OSVs to keep orbits would be better. The maximum value of the formula \( \min m_{\text{remain}}^i / m \) is \( m_{\text{keep}} / m \), which the value is less than 1. In optimization index \( B \), optimization \( B_2 \) is the main optimization index. The value of increasing or decreasing one satellite (the definition of optimization index \( B_2 \)) 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, \( O_1, O_2, \cdots, O_N \) are \( N \) OSVs, and \( S_i^j \) refers to the \( j^{th} \) satellite served by the \( i^{th} \) OSV. The value of \( S_i^j \) 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 as shown by:

\[
\max f(P_{S_i}) = \sum_{i=1}^{N} \sum_{j=1}^{M} x_{ij} + \min \frac{m_{\text{remain}}}{m} . \tag{10}
\]

In Formula (10), \( P_{S_i} \) 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]

\[
\begin{align*}
  v_{i}^{h+1} &= \omega \cdot v_{i}^{h} + c_1 \cdot r_1 \cdot (P_{g}^{h} - x_{i}^{h}) + c_2 \cdot r_2 \cdot (P_{i}^{h} - x_{i}^{h}), \\
  x_{i}^{h+1} &= x_{i}^{h} + v_{i}^{h+1} .
\end{align*}
\tag{11}
\]

In Formula (11), \( P_{g}^{h} \) is the individual extreme in the \( h^{th} \) iteration, \( P_{i}^{h} \) 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 \((h+1)^{th}\) 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 \((h+1)^{th}\) iteration, \( \omega \) 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 are written as follows:

\[
P_{S_i}^{h+1} = c_2 \otimes F_3 \left( c_1 \otimes F_2 \left[ \omega \otimes F_1 \left( P_{S_i}^{h}, P_{g}^{h} \right), P_{i}^{h} \right], P_{i}^{h} \right) . \tag{12}
\]

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

\[
\begin{align*}
  \Psi_{i}^{h} = \omega \otimes F_1 \left( P_{S_i}^{h} \right), & \quad \text{rand}() < \omega, \quad \text{rand}() \geq \omega, \quad \tag{13} \\
  \Phi_{i}^{h} = c_1 \otimes F_2 \left( \Psi_{i}^{h}, P_{i}^{h} \right), & \quad \text{rand}() < c_1, \quad \text{rand}() \geq c_1, \quad \tag{14} \\
  P_{S_i}^{h+1} = c_2 \otimes F_3 \left( \Phi_{i}^{h}, P_{i}^{h} \right), & \quad \text{rand}() < c_2, \quad \text{rand}() \geq c_2. \quad \tag{15}
\end{align*}
\]

\( \Psi_{i}^{h} \) is the value of the \( i^{th} \) particle in the \( h^{th} \) iteration after the calculation of \( \omega \otimes F_1 \left( P_{S_i}^{h} \right) \). The formula \( \text{rand}() \) means that a real number between 0 and 1 is randomly generated. When the condition \( \text{rand}() < \omega \) is met, the formula \( F_1 \left( P_{S_i}^{h} \right) \) 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 \( \text{rand}() \geq \omega \) is met, the formula \( P_{S_i}^{h} \) is executed which means that the coding values on the \( i^{th} \) particle are unchanged.

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

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

The implementation steps of the DPSO algorithm are as follows.

**Step 1.** Set parameters: the population size \( M \) 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 \( \omega \), 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 \( f(P_{\text{g}h}) \) 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 \( f(P_{\text{g}h}) \) 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 \( |P_{g}^{h} - P_{g}^{h-1}| \leq 10^{-6} \) 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 \( \text{ceil}() \) means rounding up to the nearest integer. After the count of OSVs is determined by Formula (16), the orbit element orbit radius \( (r) \), orbit eccentricity ratio \( (e) \), orbit inclination angle \( (i) \), right ascension of the ascending node \( (\Omega) \), argument of perigee \( (\alpha) \), 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 \( J_2 \)).

\[
\text{max} \ J_2 = \text{max} \ {\min J_1} = \text{max} \ {\min |f(X)|}. \tag{17}
\]

In Formula (17), \( f(X) \) 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 \( \delta t \) and the convergence precision \( \epsilon \) are set.

**Step 2.** The temporary point \( T_i \) is determined through

\[
T_i \approx \begin{cases} 
B_i + \delta t, & J_1(B_i + \delta t) < J_1(B_i), \\
B_i - \delta t, & J_1(B_i - \delta t) < J_1(B_i) \leq J_1(B_i + \delta t), \\
B_i, & J_1(B_i) < \min(J_1(B_i + \delta t), J_1(B_i - \delta t)).
\end{cases} \tag{18}
\]

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 \( B_{i+1} \). \( t^\text{th} \) 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 \(i^\text{th}\) 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 \(i^\text{th}\) vector. If the condition \( J_1(T_i) \geq J_1(B_i) \) is met, the \((i+1)^\text{th}\) vector is half of the \(i^\text{th}\) 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 \( \delta t \) 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 \( \omega \) is bigger, the global search capability of the PSO algorithm will be stronger. If the inertia weight \( \omega \) 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 \( \omega \) 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 \( \omega \), which can balance the global search capability and local search capability.
The essence of the FAPSO algorithm is updating the inertia weight $\omega$ through the fuzzy controller. The updated inertia weight $\omega_{\text{updated}}$ is determined through the fuzzy controller in three steps.

$$f_{\text{fit}} = \frac{f_{\text{fit}} - f_{\text{fitmin}}}{f_{\text{fitmax}} - f_{\text{fitmin}}}$$  \( (20) \)

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

In Formula (20), $f_{\text{fit}}$ is the fitness function value of one particle in the present iteration, $f_{\text{fitmin}}$ is the minimum fitness function value in the present iteration, and $f_{\text{fitmax}}$ 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].

$$f_s = \begin{cases} 1, & x < a_S, \\ \frac{b_S - x}{b_S - a_S}, & a_S \leq x \leq b_S, \\ 0, & x > b_S \end{cases}$$  \( (21) \)

$$f_M = \begin{cases} 0, & x < a_M, \\ 2\frac{x - a_M}{b_M - a_M}, & a_M \leq x \leq \frac{a_M + b_M}{2}, \\ 2\frac{b_M - x}{b_M - a_M}, & \frac{a_M + b_M}{2} \leq x \leq b_M, \\ 0, & x > b_M \end{cases}$$  \( (22) \)

$$f_L = \begin{cases} 0, & x < a_L, \\ \frac{x - a_L}{b_L - a_L}, & a_L \leq x \leq b_L, \\ 1, & x > b_L \end{cases}$$  \( (23) \)

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...
two input variables \( \omega \) or \( \overline{\mu} \) are within the range of fuzzy language “small.” \( a_M \) and \( b_M \) are the constants when two input variables \( \omega \) or \( \overline{\mu} \) are within the range of fuzzy language “medium.” \( a_L \) and \( b_L \) are the constants when two input variables \( \omega \) or \( \overline{\mu} \) are within the range of fuzzy language “large.”

Step 2. The values of two input variables \( \omega \) and \( \overline{\mu} \) 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

\[
\mu_{R(X,Y)}(x,y) = \mu_A(x) \land \mu_B(y). \tag{24}
\]

In Formula (24), \( x \) is the input variable \( \omega \), \( y \) is the input variable \( \overline{\mu} \), \( X \) 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 “\( \land \)” 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.

\[
\nu_{\text{out}} = \frac{\int_{-\infty}^{\infty} \nu \mu_{\nu}(\nu) d\nu}{\int_{-\infty}^{\infty} \mu_{\nu}(\nu) d\nu}. \tag{25}
\]

Step 3. The fuzzy control variable \( \Delta \omega \) is translated into the precise value after the defuzzification procedure. The precise value \( \nu_{\text{out}} \) 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), \( \mu_{\nu}(\nu) = \mu_{R(X,Y)}(x,y) \), \( V \) is the area of the curve \( f(\Delta \omega) = \mu_{R(X,Y)}(x,y) \) in Step 2 and the \( x \)-coordinate, and \( \nu \) is the \( x \)-coordinate value. The updated inertial weight \( \omega_{\text{updated}} \) is written as

\[
\omega_{\text{updated}} = \omega + \nu_{\text{out}}. \tag{26}
\]

The implementation steps of the FAPSO algorithm are as follows.

Step 1. Set parameters: the population size \( M_p \) of the particle swarm, the count \( N \) of OSVs, the maximum number of iterations \( T_C \), the inertia weight \( \omega \), the cognitive coefficient \( c_1 \), the social coefficient \( c_2 \), and \( h = 1 \). The coding of the \( i^{th} \) particle is \( P_i = [x_{i,1}^1, \cdots, x_{i,16}^{10}, x_{i,2}^{11}, \cdots, x_{i,2}^{N}, x_{i,3}^1, \cdots, x_{i,3}^{N}, \cdots, x_{i,6}^1, \cdots, x_{i,6}^{N}] \), the expression \( x_{i,k}^l \) 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 \( A, e, I, \omega, \alpha, \) 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 \( \omega \) and the normalized fitness function values \( \overline{\mu} \) in the \( h^{th} \) iteration are calculated and the updated inertial weight \( \omega_{\text{updated}} \) is determined through the fuzzy controller in three steps. \( h = h + 1 \). The updated inertial weight \( \omega_{\text{updated}} \) 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 \((|P_g^h - P_g^{h-1}| \leq 10^{-6}) \) 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 \( (A, \epsilon, I, \Omega, \alpha, \text{and} f) \) of the OSVs determined according to the orbit elements of all satellites are \( [6700 \text{ km}, 11000 \text{ km}], [0, 1], [0', 180°], [0', 360°], [0', 360°], \text{and} [0°, 360°] \), 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’ maneouvre 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) | $m_{\text{remain}}$ (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 | —         | —         | —         | 164.904                  |

Figure 5: Parking orbits of all OSVs.

Figure 6: Transfer orbit of O1 serving S8, S1, and S18.
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 $m_{\text{keep}}$ 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 $\omega$ 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 $\omega$ 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 $A$, $e$, $I$, $\Omega$, $\alpha$, 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 $\delta t$ of the pattern search is $(24 \times 60 \times 60)$ seconds and the time interval $T$ is
(2 × 365 × 24 × 60 × 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 10th 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 J2), the time to obtain max J2, 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 J2) is after 9336.450 hours. O1, O2, O4, O6, and O7
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 $m_{\text{remain}}$ is the weight of remaining fuel after the $i^{th}$ 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.
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.

![Figure 13: Transfer orbit of O8 serving S2 and S4.](image)

![Figure 14: The convergence curves of the FAPSO algorithm and other algorithms.](image)
Notations

\( t \): Uncertain time

\( T \): Time interval

\( t_1, t_2, \ldots, t_{2n-1} \): Manoeuvre times

\( t_2, t_4, \ldots, t_{2n} \): Rendezvous times

\( T_{\text{max}} \): Maximum time interval for finishing the on-orbit mission

\( t_{\text{serve}} \): Time interval for the OSV serving one satellite

\( r_i \): Position vector at point \( P_i \)

\( \nu_i \): Velocity of the original orbital at point \( P_i \)

\( \nu_2 \): Velocity vector, respectively, at point \( P_2 \)

\( \nu_i \): Velocity of the target orbit at point \( P_i \)

\( \Delta v_{i1} \): The 1st velocity increment of the OSV serving the \( i \)th satellite at the manoeuvre time

\( \Delta v_{i2} \): The 2nd velocity increment of the OSV serving the \( i \)th satellite at the rendezvous time

\( m \): Weight of a single OSV

\( m_{\text{fuel}} \): Weight of the fuel carried by the OSV

\( m_{\text{else}} \): Weight of the other parts

\( m_{\text{remain}} \): Weight of the remaining fuel after the on-orbit mission is achieved

\( \Delta m \): Weight of the consumed fuel

\( m_{\text{keep}} \): Weight of fuel used to keep the orbit

\( m_{\text{remain}}^{i-1} \): Weight of the remaining fuel after a single OSV serves \( i-1 \) satellites

\( m_{\text{remain}}^i \): Weight of the remaining fuel after the \( i \)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

\( P_{i}^{h} \): Fitness function value of the \( i \)th particle in the \( h \)th iteration

\( P_{i}^{h+1} \): Fitness function value of the \( i \)th particle in the \((h+1)\)th iteration

\( \Psi_i^h \): Individual extreme in the \( i \)th iteration

\( \Phi_i^h \): Global extreme in the \( h \)th iteration

\( \max f(X) \): Optimization index \( B \)

\( \text{fit} \): Fitness function value of one particle in the present iteration

\( \text{fit}_{\text{min}} \): Minimum fitness function value in the present iteration

\( \text{fit}_{\text{max}} \): Maximum fitness function value in the present iteration

\( \psi_i^h \): Value of the \( i \)th particle in the \( h \)th iteration after the calculation of \( \omega \otimes F_1 (P_{i}^{h}) \)

\( \phi_i^h \): Value of the \( i \)th particle in the \( h \)th iteration after the calculation of \( c_1 \otimes F_2 (\Psi_i^h, P_{i}^{h}) \)

\( \omega \): Inertia weight

\( x \): Input variable \( \omega \)

\( y \): Input variable \( \frac{\text{fit}}{\text{fit}_{\text{max}}} \)

\( X \): 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 *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. Mohammad-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 *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.