aerospace

MDPI

Article

Hierarchical Optimization Algorithm and Applications of Spacecraft Trajectory Optimization

Hanqing He, Peng Shi * and Yushan Zhao

School of Astronautics, Beihang University, Beijing 102206, China; hanqing_he@buaa.edu.cn (H.H.); yszhao@buaa.edu.cn (Y.Z.)
* Correspondence: shipeng@buaa.edu.cn

Abstract: The pursuit of excellent performance in meta-heuristic algorithms has led to a myriad of extensive and profound research and achievements. Notably, many space mission planning problems are solved with the help of meta-heuristic algorithms, and relevant studies continue to appear. This paper introduces a hierarchical optimization frame in which two types of particles—B-particles and S-particles—synergistically search for the optima. Global exploration relies on B-particles, whose motional direction and step length are designed independently. S-particles are for fine local exploitation near the current best B-particle. Two specific algorithms are designed according to this frame. New variants of classical benchmark functions are used to better test the proposed algorithms. Furthermore, two spacecraft trajectory optimization problems, spacecraft multi-impulse orbit transfer and the pursuit-evasion game of two spacecraft, are employed to examine the applicability of the proposed algorithms. The simulation results indicate that the hierarchical optimization algorithms perform well on given trials and have great potential for space mission planning.

Keywords: meta-heuristics; hierarchical optimization algorithm; multi-impulse orbit transfer; spacecraft pursuit-evasion game

Citation: He, H.; Shi, P.; Zhao, Y. Hierarchical Optimization Algorithm and Applications of Spacecraft Trajectory Optimization. *Aerospace* **2022**, *9*, 81. https://doi.org/10.3390/aerospace9020081

Academic Editors: Lorenzo Casalino and Mikhail Ovchinnikov

Received: 23 November 2021
Accepted: 29 January 2022
Published: 3 February 2022

1. Introduction

In the last twenty years, the rapid development of meta-heuristic optimization algorithms has promoted the extraordinary progress of widespread engineering applications, including variable selection in chemical modeling [1], pattern recognition [2], path planning of UAVs [3], feature selection [4] and data clustering [5]. Doubtlessly, the powerful capabilities of parameter optimization are beneficial to space missions planning, such as spacecraft rendezvous trajectory optimization [6], interplanetary trajectories by multiple gravity-assist [7], agile satellite constellation design [8] and spacecraft attitude maneuver path planning [9]. In order to effectively apply the meta-heuristic algorithms to the space mission planning, on the one hand, constructing an appropriate optimization problem model is of vital importance. On the other hand, it is valuable to improve the performance of algorithms, which is the focus of this paper. Compared with traditional mathematical programming methods, meta-heuristic algorithms attract the attention of a large number of scholars due to four characteristics: simplicity, flexibility, derivation-free mechanism and local optima avoidance [10]. Relevant research can be divided into three types: proposing new optimization algorithms, improving existing algorithms and applying existing algorithms to practical problems. Apparently, new algorithms are most noteworthy for their new mechanisms to provide inspiration for other methods.

To design a highly applicable meta-heuristic algorithm, researchers have abstracted phenomena and laws in the real world into mathematical descriptions, from which a great number of algorithms were born. Genetic Algorithm (GA) [11] imitates Darwinian evolution theory and is a pioneer of multitudinous evolutionary optimization algorithms, including Evolutionary Programming (EP) [12] and Differential Evolution (DE) [13]. Typical

mechanisms in this type of algorithm include selection, crossover and mutation, which have been extensively embedded in other algorithms to improve their performance [14]. To date, improved evolutionary algorithms are still studied by researcher [15]. The hybridization of evolutionary algorithms and local search operators produces the memetic algorithm (MA) [16], in which the local search is used to improve search efficiency. The efficiency and effectiveness of the local search mechanism depends on three principal components: the pivot rule, the iteration condition and the neighborhood generating function. The strategy of local search can be fixed or adaptive, according to the performance of multiple operators [17]. The primary characteristic of MA is the evolutionary operators used in global exploration, and multitudinous algorithms have been proposed to design and combine appropriate local search strategies to better solve the problems.

Individuals in evolutionary algorithms are relatively independent, and the connection depends only on crossover between individuals. Swarm intelligence (SI) optimization can strengthen and consolidate the connection. Particle Swarm Optimization (PSO) [18] is one of the most popular SI algorithms, which is inspired by the behavior of birds flocking in nature. In PSO, every particle makes decisions about its movement by self-cognition and social cognition. Self-cognition is the historical best position of a particle, and social cognition means the position of the current best particle. Improvement of PSO is proposed by many researchers to attempt to eradicate the shortcomings of premature convergence and lack of dynamicity [19], and to enlarge the applying scope [20]. A review of PSO was conducted by [21]. A mechanism of learning from the best particle has been introduced to many SI algorithms, such as Firefly Algorithm (FA) [22], Grey Wolf Optimizer (GWO) [10], Harris Hawk Optimization (HHO) [23], etc. For more detailed information about reviews on swarm intelligence and evolutionary algorithms, [24] is recommended. There are also many optimization algorithms designed according to physical phenomena and mathematical operations, including Simulated Annealing (SA) [25], Gravitational Search Algorithm (GSA) [26], Sine Cosine Algorithm (SCA) [27] and Arithmetic Optimization Algorithm (AOA) [28]. While there are already many optimization algorithms, there are still new optimization algorithms continually being proposed. The No Free Lunch (NFL) theorem is an explanation for this phenomenon [29], which proves that no algorithm is able to handle all problems well. Therefore, this motivated us to propose a new method for optimization.

The optimal solution for space mission planning problems is usually difficult to find, due to the complex landscape of the search space, which ultimately leads to algorithms obtaining suboptimal solutions. The reason is that the process of solution search has two phases: exploration for global search and exploitation for local search. Considering the contradiction between the two phases, existing algorithms usually switch from exploration to exploitation over the iterations, and strive to balance the ratio of two phases to increase the search efficiency. However, the algorithms pay heavy penalties for extricating from the vicinity of suboptimal solutions. If an algorithm can simultaneously perform both exploration and exploitation over the iterative search, the search efficiency will be improved. The goal of this paper is to solve this problem to a certain extent.

In this paper, a new swarm intelligence two-hierarchy optimization frame (HOF) is proposed. The core idea of HOF is the rational allocation of limited computing resources, namely the number of function evaluations. Motivated by the trends of the local search operations incorporated in memetic algorithms, we designed the HOF by endowing the best individual with additional function evaluations as a reward to enhance the search ability of the best individual compared with others. The strategy of HOF is to generate two types of particles, B-particle and S-particle, to search the variable space in two hierarchies, respectively. Therefore, the total number of function evaluations is determined according to the iteration numbers of two hierarchies, the B-particle number and the S-particle number. In the search of the first hierarchy, a certain number of B-particles explore the space iteratively. The motion direction and step length of B-particles are determined separately to improve the designability of algorithms and increase the exploration efficiency of B-particles. In each iteration of B-particles, the best B-particle has additional function evaluations pro-

vided by the local search of the second hierarchy, in which a certain number of S-particles iteratively exploit the neighborhood, and the best result is returned to update the position of the best B-particle. The number of function evaluations is a constant predefined and identical to the total number of generated S-particles. Two specific algorithms, HOA-1 and HOA-2, are designed based on HOF, and the difference lies in the local search methods of S-particles. In both algorithms, B-particles, except the best one, iteratively search the space by referring to the best two B-particles. In HOA-1, S-particles perform a pattern search by iteratively searching the space dimension by dimension, and the search step length in each dimension is a uniformly distributed random number with stepped decreasing bounds with the increase in the iteration number of first hierarchy. In each iteration of the second hierarchy, S-particles in HOA-2 are generated simultaneously based on a set of Gaussian distributions, which are iteratively updated according to the positions of the current best two function values.

To test the effectiveness of algorithms, the benchmark functions test is a popular method. However, some existing algorithms perform well in solving benchmark functions whose solutions are the origin of the search space, and an obvious performance degradation appears when the solutions deviate a little from the origin. Considering this unreasonable phenomenon, twenty-three variant benchmark functions are designed. In contrast to regular benchmark functions repeated solving, we set a random position deviation to the solution of the benchmark function, while keeping the minimum value invariant. Therefore, these variant benchmark functions were solved by the two proposed algorithms and seven existing algorithms repeatedly. Besides the benchmark functions test, two spacecraft trajectory optimization problems were solved by proposed algorithms and compared algorithms. The first problem was the spacecraft multi-impulse orbit transfer between two coplanar orbits, and the target was to find the optimal velocity impulse vectors and corresponding time. The second problem is the two-spacecraft pursuit-evasion game solved by the multiple shooting method, and the target is to find a set of appropriate costate variable initial values.

The rest of this paper is arranged as follows: Section 2 gives a statement about the optimization problem to be solved. Section 3 describes HOF and two algorithms systematically. The performance tests of proposed algorithms on benchmark functions and trajectory optimization problems are respectively presented in Sections 4 and 5. The conclusion is given in Section 6 ultimately.

2. Optimization Problem Formulation

Optimization problems cover a wide range of subproblem types, which can be single-objective or multi-objective, unconstrained or constrained, static or dynamic. The focus of this work is to solve the static unconstrained single-objective optimization problem (SUSOP), since other types of problems can be studied by introducing more techniques and mechanisms into the achievement of SUSOP. Stochastic/heuristic optimization techniques have been extensively employed to handle optimization problems, and explicit reviews in [10,27] may help interested readers gain a comprehensive understanding of the development in this field.

This section gives a brief description and definition of the problem to be studied in the following work. Usually, a minimization problem is formulated to represent the optimization problem, as follows:

$$\begin{aligned} &\text{Minimize}: f(X) \\ &\begin{cases} X = (x_1, x_2, \ldots, x_{N-1}, x_N)^T \\ lb_i \le x_i \le ub_i,\ i = 1, 2, \ldots, N \\ Ub = (ub_1, ub_2, \ldots, ub_N)^T \\ Lb = (lb_1, lb_2, \ldots, lb_N)^T \end{cases} \end{aligned} \tag{1}$$

where lb_i and ub_i are boundary values of the ith element of state variable X due to the restrictions on an N-dimension search space in numerical calculation. The continuity of function f means that a best solution must exist in the search space, even though the global optima may not be located in this space, especially for those problems in which it is difficult to stipulate the search space, for instance, when it is necessary to search for the initial values of costate variables in two-point boundary problem solving.

To find the best solution of the SUSOP, abundant algorithms have been proposed. Meta-heuristic algorithms are one kind of method quite popular over recent decades. A typical feature of meta-heuristics is the random factor, which leads meta-heuristics to receive different results so as to increase the probability of finding the optimal solution. A meta-heuristic algorithm (MHA) can be formulated as follows:

$$X^* = \mathrm{MHA}(f(X), X_0, \{P\}) \tag{2}$$

where $f(X)$ is the optimization function with a given search space, X_0 is the initial values, {P} is the parameter set including constant and varied parameters, and X^* is the solution of $f(X)$ obtained by MHA. The performance of MHA is significantly influenced by the values of {P}. However, the focus of this work is to design the search strategy, while the values of parameters are intuitively set and manually adjusted by the simulation results in the following paper.

3. Hierarchical Optimization Algorithm

The structure of HOF is outlined in this section. Subsequently, two specific algorithms, HOA-1 and HOA-2, are provided and their differences are discussed.

3.1. Hierarchical Optimization Frame

In this paper, we used particles to represent the individuals in SI algorithms. For SI algorithms, a popular idea is to use the position of the current best particle (CBP) to guide other particles' motion, which makes the exploration and exploitation in the search space more effective and efficient. However, CBP itself seldom obtains benefit from this strategy. To alleviate this limitation, HOF was proposed to enhance the search ability of CBP by performing a local search and giving it additional function evaluations. HOF includes two hierarchies: H1 is the first hierarchy and H2 is the second hierarchy. In H1, a certain number of particles called B-particles are used to iteratively search for the global optimum. Therefore, the CBP in HOF is the current best B-particle. In each H1 iteration, B-particles cooperate to iteratively update their positions based on the current information of search results. Generally, only one function evaluation is conducted to complete the position update of each B-particle except CBP, whose position update depends on the local search of H2 iterations and requires more function evaluations. In H2, particles of another type called S-particles, initially distributed in the neighborhood of CBP, are assigned to iteratively exploit the space. After a predefined constant number of S-particles are randomly generated, the position of the best S-particle becomes the updated position of CBP based on the greedy strategy. The iterations of two hierarchies are executed in turn until the maximum number of H1 iterations is reached. Note that B-particles converge as the H1 iterative search progresses, while S-particles dispersedly exploit the local space from the initial position of CBP. I_b and I_s are the maximum number of iterations in H1 and H2, which means that I_s times H2 iterations proceed after each H1 iteration, and the stopping criterion is I_b times H1 iterations have been performed. The total number of function evaluations is determined according to the iteration numbers I_b and I_s, the B-particles number and the S-particles number. The whole frame is modularized and illustrated by Figure 1, in which the orange blocks represent adjustable sub-algorithms. A 2-D sphere function was used to show the CBP's update process in one H1 iteration with its subsidiary H2 iterations. In Figure 2, Figure 2a shows the positions of eight B-particles, in which the CBP is represented by a red marker, while other B-particles are blue markers. Four S-particles, marked by pentagrams, were used to perform the local search in the neighborhood of the CBP, and

their initial positions and final positions are respectively given in Figure 2b,c. The best S-particle was selected to update the CBP's position after H2 iterations, and is denoted by the green marker in Figure 2d.

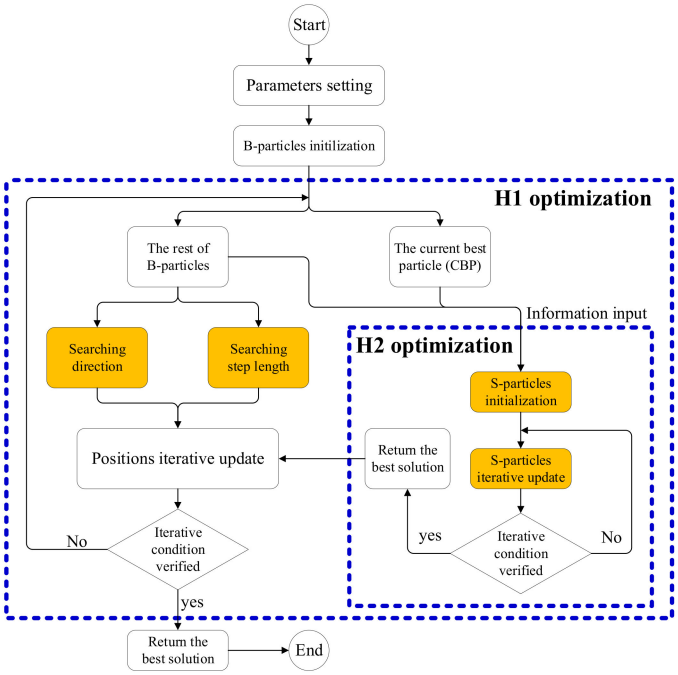


Figure 1. Flowchart of HOF.

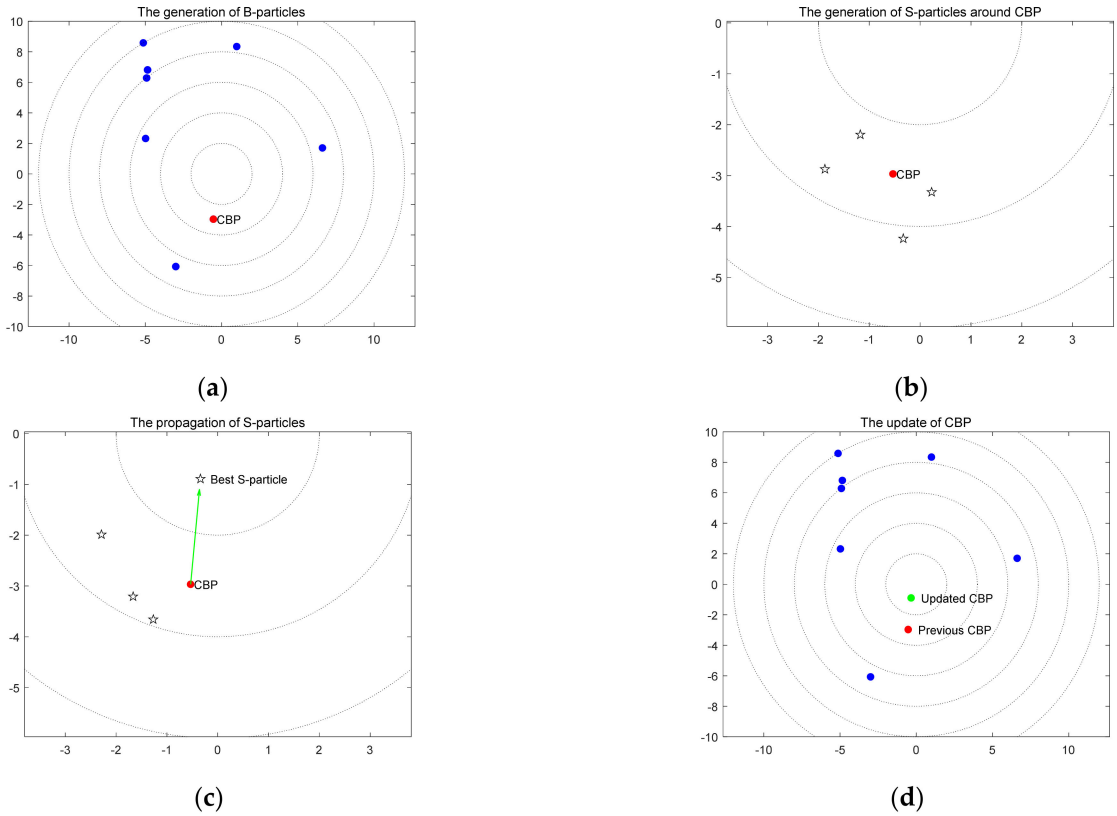


Figure 2. Update process of CBP in HOF.

There is no specific formula given in the flowchart in Figure 1. To design specific algorithms, we designed an iterative method for B-particles used in this paper, while other methods are also applicable. In H1, we culled the best two B-particles in each iteration and set them as two leaders, $L1$ and $L2$, to guide the motion of other B-particles in the next iteration, except for $L1$'s position, which was determined by the result of H2 iterations. Each SI algorithm has its own way of deriving particle propagation, including two aspects, the direction and the step length. The velocity computation of PSO provides a particle's motional direction and step length simultaneously. In comparison, the direction and step length of bacteria are computed respectively in Bacterial Foraging Algorithm [30], in which the direction is a unit vector randomly produced in the searching space, and the step length is a small predetermined value to meet search accuracy. Both algorithms have respective advantages and limitations, which inspired us to separate the determination of searching direction and step length to strengthen the algorithm's flexibility and adaptability for different optimization problems.

Since the search direction and step length are calculated respectively, we used positions of $L1$ and $L2$ to derive a particle's direction update formula by

$$D_i^n = \frac{c_1(L1^n - B_i^n) + c_2(L2^n - B_i^n)}{\left\| c_1(L1^n - B_i^n) + c_2(L2^n - B_i^n) \right\|} \tag{3}$$

where B_i^n indicates ith B-particle's position in nth iteration in H1, $L1^n$ and $L2^n$ represent the positions of $L1$ and $L2$ in nth iteration respectively and the parameters c_1 and c_2 control the weights of $L1$ and $L2$ to B_i^n. The symbol $\|\cdot\|$ means the Euclidean norm of a vector. With the inconsistency of $L1$ and $L2$, the denominator of D_i^n is always larger than 0, which averts the singularity. Assuming $L1^n$ exerts more influence on B_i^n than $L2^n$, we set $c_1 > c_2 > 0$. The step length of B_i^n is given by

$$deltaB_i^n = \alpha(\|L1^n - B_i^n\| + \|L2^n - B_i^n\|)/2 \tag{4}$$

where α is the step ecoefficiency between 0 and 1 to directly tune the searching range of B_i^n. In total, the updating equation of B_i^n in H1 iteration is constructed as follows:

$$B_i^{n+1} = B_i^n + deltaB_i^n \cdot D_i^n \tag{5}$$

note that the update mechanism of B-particles is deterministic, due to the constant values of c_1, c_2 and α, which is not best, but effective. Further study on the autonomous adjustment of these parameters can be conducted by introducing randomness or adaptive laws.

$L1^n$ is actually the position of CBP in nth H1 iteration. Different ways to generate and update S-particles can construct many specific algorithms, one of which may better solve the focused problem. Two hierarchical optimization algorithms are introduced in the next subsection, and their difference lies in the local search method of S-particles.

3.2. Two Hierarchical Optimization Algorithms

Considering the performance differences caused by the different searching methods of S-particles, two hierarchical optimization algorithms, HOA-1 and HOA-2, are proposed to validate the effectiveness of HOF.

3.2.1. Formula of HOA-1

The searching method of S-particles can be divided into two steps: initialization and iterative update. The initialization is to determine the spatial domain and the position distribution law. The first step is to set a spatial domain of initial S-particle positions. A gradually shrinking domain makes a fine effect, because S-particles are used to search the

neighborhood of CBP. In order to simplify the algorithm, we designed a function to control the boundary of this domain in Equation (6).

$$deltaS^n = \frac{(Ub - Lb)}{F(n/T)},\ n = 1, 2, \ldots, I_b \tag{6}$$

where $F(\cdot)$ is an operator to tune the range of $deltaS^n$ in nth H1 iterations, Ub and Lb are the boundary of the search space defined by (1), and T is a parameter to control the phased variation of search precision. In HOA-1, $F(\cdot)$ is defined in (7).

$$F(n/T) = 10^{-\lfloor n/T \rfloor - c} \tag{7}$$

The symbol $\lfloor \cdot \rfloor$ indicates rounding down the variable. This function means the domain stays invariant for T times H1 iterations, and shrinks to one tenth in the next T iterations, and c is a constant used to determine the initial search precision. Generally speaking, we suppose that a better position exists within a hypercube centered on CBP, and take $deltaS^n$ to set the length of its edges.

In HOA-1, the number of S-particles is equal to the dimension N of the problem to be solved, which results from the idea that changing the value of CBP's position dimension by dimension serves to precisely search the space around CBP.

In H2 iterating process, the reference point P_r is used to update the positions of S-particles, and the initial P_r is the CBP of the B-particles in the current H1 iteration. Once a better position is found, it becomes the new P_r of the next H2 iteration. Let S_j^m be the position of jth S-particle in mth H2 iteration. The updating equation of S_j^m is constructed as follows:

$$S_j^m = P_r + r_s \cdot I_{N,j} \cdot deltaS^n(j),\ m = 1, 2, \ldots, I_s \tag{8}$$

where $I_{N,j}$ is the jth column vector of a N-order identity matrix I_N, $deltaS^n(j)$ is the jth element of $deltaS^n$, and r_s is a random number uniformly distributed in the interval of [–1,1]. After H2 iterations, the final P_r is output to renew the position of CBP if its function value is smaller. The HOA-1 algorithm is summarized in Algorithm 1.

Algorithm 1 Pseudo code of HOA-1.

Generate initial B_i^1 (i = 1, 2, . . . , p)
Calculate the fitness of each B-particle f(B_i^1)
Choose $L1$ and $L2$
Starting H1 iterations:
while 1 ($n < I_b$)
  for 1 B_i^n
    if 1 B_i^n doesn't equal $L1^n$
      Update the position by (5)
    else
      Starting H2 iterations:
      Input CBP's position as P_r and fitness as f_r and $deltaS^n$ by (6)
      while 2 ($m < I_s$)
        for 2 S_j^m(j = 1, 2, . . . , N)
                Generate S_j^m by (8)
        if 2 f(S_i^m) < f_r
          $P_r = S_i^m$
        end if 2
        end for 2
      end while 2

Output P_r as CBP's updating position
end if 1
end for 1
Update $L1$ and $L2$ according to $f(B_i{}^{n+1})$
end while 1
return the final $L1$

3.2.2. Formula of HOA-2

In HOA-1, the way S-particles search for a better solution is by changing only one element of P_r in a given domain to generate a new S-particle that is uniformly distributed in the hypercube. To design a different search strategy, a new algorithm HOA-2 is proposed in which the number of S-particles can be specified arbitrarily, every element of each S-particle is obtained from a Gaussian distribution, and all elements are changed simultaneously. To determine a suitable Gaussian distribution, the mean value μ and the standard deviation σ should be defined. Thus, the positions of $L1$ and $L2$ are input to the H2 optimization, and μ and σ can be given as follows:

$$\begin{cases} \mu_n^1 = L1^n \\ \sigma_n^1 = abs(L1^n - L2^n) \end{cases} \tag{9}$$

where μ_n^1 and σ_n^1 represent the first μ and σ in the H2 optimization of nth H1 iteration, respectively. The operator $abs(\cdot)$ is used to replace all elements of the vector with their absolute values. This specification stems from the assumption a better position exists closer to $L1$ than $L2$. Given that the range of this parameter setting is sometimes too large, another specification in Equation (10) is proposed, which is also subject to the above assumption.

$$\begin{cases} \mu_n^1 = (2L1^n + L2^n)/3 \\ \sigma_n^1 = abs(L1^n - L2^n)/3 \end{cases} \tag{10}$$

The difference between Equations (9) and (10) is illustrated in Figure 3, in which the curves are the probability distribution density functions of two ways. It shows the interval constrained by Equation (9) is obviously larger. Due to the high probability of the generation of new S-particles between $L1$ and $L2$ by using Equation (10), this specification is preferred when the best solution is encircled by S-particles.

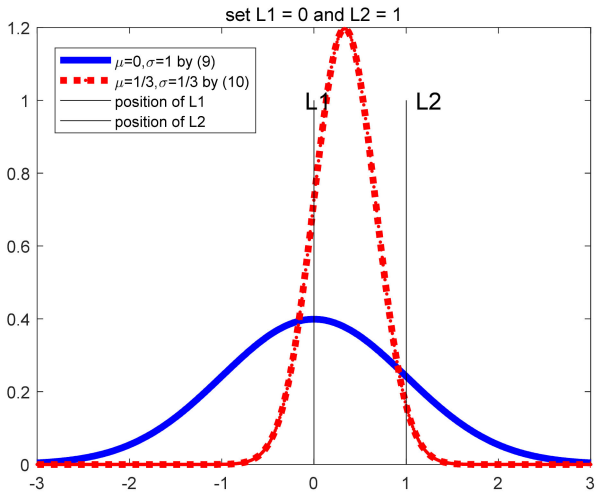


Figure 3. The difference between two specifications of μ and σ.

To guarantee the validity of this estimation for better solutions, μ and σ are updated after every H2 iteration, which leads to the necessity of selecting and updating the best

two S-particles, namely $SL1$ and $SL2$. Using $SL1$ and $SL2$ to severally replace $L1$ and $L2$ in Equations (9) or (10) can update μ and σ in each H2 iteration.

When the iterations in H1 proceed, if no other B-particle takes over these two particles, the difference between $L1$ and $L2$ becomes small, which makes the search space in H2 too narrow for S-particles to help CBP find a better position. With that in mind, a lower bound for σ is introduced to prevent S particles from being limited to a cramped space search. As with the restriction on $deltaS^n$, Equation (11) gives the definition of σ and $\sigma_{\min}$.

$$\sigma_{\min} = \frac{(Ub - Lb)}{F(n/T)},\ \sigma_n^m = \begin{cases} abs(SL1^m - SL2^m), abs(SL1^m - SL2^m) \geq \sigma_{\min} \\ \sigma_{\min},\ \text{else} \end{cases} \tag{11}$$

where σ_n^m is σ in mth H2 iteration of nth H1 iteration, and $SL1^m$ and $SL2^m$ are positions of $SL1$ and $SL2$ in mth H2 iteration, respectively. Therefore, based on the setting of μ and σ in Equation (9), the updating equation of jth S-particle S_j^m in mth H2 iteration is constructed as follows:

$$S_{j,\,k}^{m+1} = Z_k,\ Z_k \sim N(SL1_k^m, \sigma_n^m(k)) \tag{12}$$

where the subscript k means kth element of a vector and $\sigma_n^m(k)$ is kth element of σ_n^m. Likewise, the setting of μ and σ based on Equation (10) can be derived, but is omitted here. The HOA-2 algorithm is summarized in Algorithm 2. To maintain the relative independence between H1 and H2, only $SL1$'s final position is output to update CBP's position.

Algorithm 2 Pseudo code of HOA-2.

```
Generate initial B_i^1 (i = 1, 2, ... , p)
Calculate the fitness of each B-particle f(B_i^1)
Choose L1 and L2
Starting H1 iterations:
while 1 (n < I_b)
  for 1 B_i^n
    if 1 B_i^n does not equal L1^n
      Update the position by (5)
    else
      Starting H2 iterations:
      Input the position and fitness of L1 and L2
      while 2 (m < Is)
        for 2 Sjm(j = 1, 2, ... , q)
                    Generate Sjm by (12)
          Update the position and fitness of SL1 and SL2
          Update μ and σ by (9) and (11).
        end for 2
      end while 2
      Output SL1q as CBP's updating position
    end if 1
  end for 1
  Update L1 and L2 according to f(B_i^{n+1})
end while 1
return the final L1
```

HOA-1 and HOA-2 share the same H1 and differ from the search mechanism in H2, which manifests more optimization algorithms that can be proposed based on HOF. By introducing more mechanisms into HOF, such as evolutionary operators, simulated annealing mechanism and other effective mechanisms, a new algorithm with better performance can be accomplished.

3.3. Comparison with Other Algorithms

The concept of hierarchical optimization has been used in algorithm design and some results have been employed in single-objective optimization and multi-objective optimization. In [31], a hierarchical algorithm (HGA-PSO) was developed to construct two-layer optimization, in which the bottom layer, responsible for exploration, is based on GA, and the top layer, for exploitation, adopts PSO. The bottom layer includes many subgroups searching for the best solution independently. After each iteration in the bottom layer, some excellent individuals of every subgroup are brought together to form a new group to continue cooperative search in the top layer. Until the termination conditions are met, k individuals are selected randomly from the top layer to substitute random k individuals of every subgroup in the bottom layer before the next iteration starts. An external archive is established to store the results of the top layer search, and subgroups choose k individuals from the archive [32], which eliminates the negative influence of initialization and randomness of individuals in the top layer by outputting the final solution from the archive. Compared with the hierarchical optimization methods mentioned above, the HOF is based on a competition mechanism, which is analyzed in detail in Section 4.2.2 through simulation examples, instead of combining the best individuals from multiple subgroups to conduct a cooperative optimization. In [33], a multiagent collaborative search (MACS) method was proposed for local exploration of the subdomain generated by the branching method. Multiple agents were generated in the given subdomain and used to explore this subdomain by elaborate collaboration mechanisms. Afterwards, MACS was further developed based on a combination of Tchebycheff scalarization and Pareto dominance to solve multi-objective optimization problems [34]. Compared with MACS, the exploitation of S-particles around CBP is quite different. The most obvious difference is that S-particles locally search the space around CBP independently, while strong information interaction exists between agents in MACS. Also, a search subdomain is firstly determined, and then the population of agents is introduced and updated to search this invariant subdomain, while the local search space of S-particles varies with the position update of the current optimal solution.

Regarding the H2 optimization as the local search of H1 optimization, the frame of HOF is similar to memetic algorithms. Therefore, a further comparison of two proposed algorithms and existing algorithms is conducted from the point of view of the local search. An adaptive gradient descent-based local search was introduced in MA [35], in which the numbers of individuals considered in local search, the search steps and iterations decreased linearly during the iterations. However, the gradient calculation requires additional function evaluation to generate a new individual, which leads to the important task of determining the ratio of the function evaluations of gradient calculation to the total evaluation number. In HOA-1 and HOA-2, the local search of S-particles proceeds without gradient information, and accordingly, once function evaluation occurs when generating a new S-particle. The local search mechanism of HOA-1 can be classified as a pattern search method with random factors. The stepped decreasing upper bound of iterative step length of S-particles in each dimension is related to the iteration number of H1, and differentiates HOA-1 from other local pattern search methods, for example, the memetic algorithm combining PSO and pattern search refinement [36], and multi- coordinate search [37]. This stepped decreasing mechanism of search step guarantees the search ability of CBP, even if all B-particles converge in the early iterations of H1 and avoids the local minima.

Another point worth demonstration concerns the Gaussian distribution used in HOA-2. Kennedy proposed a Bare Bones Particle Swarms Optimization (BBPSO) in 2003 [38], in which every particle's position update is obtained by the Gaussian distribution defined by the best particle position and its historical optimal position. Then, many researchers improved the mechanism of Gaussian distribution parameter setting to increase its searching efficiency [39,40]. In these algorithms, the weight of best individual in defining the Gaussian distribution parameters is identical to other individuals, due to the balance of exploration and exploitation during the iterations. In HOA-2, the weight of $L1$ or $SL1$ is

determinately bigger than that of $L2$ or $SL2$, because it adopts the hypothesis that the solution is closer to the best particle than other particles. Understandably, this hypothesis leads to closer distances between sampled particles and the best particle, and thus causes the premature convergence. Therefore, this hypothesis has a negative effect on the exploration and is rejected by existing algorithms. However, only the best one of sampled S-particles is selected to update the position of the CBP, and other S-particles have no influence on the iterative exploration of B-particles, which allows an increase in the weight of the best particle. In HOA-2, the S-particles are simultaneously sampled based on the same set of Gaussian distributions in each H2 iteration, and these Gaussian distributions are updated according to the positions of two best S-particles $Sl1$ and $SL2$. However, each particle in BBPSO has its own set of Gaussian distributions, which means that only one particle is sampled based on these distribution functions. The multiple sampling based on the Gaussian distributions can better exploit the neighborhood of the best particle, while the randomness of single sampling brings a negative effect to the local search. The local search methods based on covariance matrix adaption also utilize multiple dimension Gaussian distribution to describe the sampling subdomain around the center of selected individuals [41]. However, this type of algorithm emphasizes the adaptive laws of the covariance matrix and step size, without considering the weights of dominant individuals.

4. Experiments on Benchmark Functions

In this section, twenty-three benchmark functions are used to validate the performance of HOA-1 and HOA-2 compared with other optimization algorithms. Then, analysis and discussion about the results are elucidated.

4.1. Explanations of Performance Test

For comprehensive inspection of optimization algorithms, twenty-three benchmark functions, which consist of three types of functions—unimodal, multimodal and fixed-dimension multimodal—were utilized in [10]. To better examine the performance of proposed HOA-1 and HOA-2, we designed variants of these twenty-three benchmark functions instead of directly using them to testify the overall effectiveness. The origin benchmark functions are listed in Tables A1–A3 in Appendix A. We introduced skills of variable substitution to shift the theoretical solution of each benchmark function in the searching space, while keeping the minima unchanged. The variable substitution can be expressed as $x_i = y_i - r_i$, where r_i is a random value in $[(19ub_i + 21lb_i)/40, (21ub_i + 19lb_i)/40]$ which covers $1/20$ of the range of ith dimension search space. Taking F1 sphere function as an example, the actual fitness function is $\sum_{i=1}^{30}(y_i - r_i)^2$, the range of each dimension is still [–100, 100], and r_i is a uniformly distributed constant in [–5, 5], which means the extreme point is no longer the coordinate origin but a random point near the origin. Solving the minima of F1 function repeatedly, the theoretical value is always 0, while the position is $R = [r_1, \ldots, r_{30}]^T$, which is determined by the results of random sampling before the optimization starts. In this way, the efficacy of algorithms emerges more clearly.

In contrast to the proposed algorithms, seven algorithms were introduced as comparison algorithms, including three classical algorithms: GA, DE and PSO, and four recently proposed algorithms: GWO, Butterfly Optimization Algorithm (BOA) [42], HHO and AOA. The parameters of these algorithms are defined in Table 1. The numbers of iterations $I_{\max}$ of all these contrastive algorithms were all set to 500, and the population size was 30. Since two hierarchies exist in HOA-1 and HOA-2, we set I_b to 100 and I_s to 4 for the sake of fairness, which guarantees a smaller number of function evaluations in HOA-1 and HOA-2 than in those of the compared algorithms. Because different benchmark functions require different search precision, the parameter T in HOA-1 and HOA-2 is defined correspondingly.

Table 1. Parameters setting.

Algorithm	Parameters
GA	$Pc = 0.8$, $Pm = 0.2$, $Pr = 1.5$
DE	$F = 0.3$, $CR = 0.2$
PSO	$\omega_{max} = 0.9$, $\omega_{min} = 0.4$, $c_1 = c_2 = 2$
GWO	$a = 2(1\text{-}i/I_{\max})$, i is the current iteration
BOA	$a = 0.1 + 0.2 \times i/I_{\max}$, i is the current iteration, $c = 0.01$, $p = 0.8$
HHO	$\beta = 1.5$, $J = 2(1\text{-}r_5)$, r_5~U(0,1)
AOA	$\alpha = 5$, $\mu = 0.5$
HOA-1	$\alpha = 0.3$, $c_1 = 1$, $c_2 = 0.3$, $c = 1$
HOA-2	$\alpha = 0.3$, $c_1 = 1$, $c_2 = 0.3$, $c = 1$

These algorithms were executed 30 times to solve each benchmark function on a laptop with an Intel(R) Core (TM) i7-10510U CPU at 1.8 GHz and 16.0 GB of RAM. All these algorithms are coded in MATLAB R2020a.

4.2. Results and Discussion

4.2.1. The Performance

The results of previous experiments are arranged meticulously, and the data are displayed in Tables 2–4, in which the best results obtained by nine algorithms for each benchmark function are presented in bold. In these three tables, ***B***, ***A*** and ***S*** respectively stand for the best result, the average value and the standard deviation. The results of benchmark function test are given in Tables 2–4, in which the minimum values of these benchmark functions are bold. In the unimodal benchmark functions test, HOA-1 and HHO both found the best solution of three of the seven functions, and showed outstanding exploitation ability, according to Table 2. Furthermore, the results of the multimodal functions test in Table 3 illustrate that the exploration ability of HOA-1 is superior to other algorithms. For local minima avoidance, the data in Table 4 demonstrate that all these algorithms performed similarly, while HOA-1, DE and PSO had a weak advantage on the others. To statistically display the performance of tested algorithms, the Friedman test [43] was used to reflect the performance, and the result is listed in Table 5, which proves the advantage of HOA-1 over other algorithms. The p-value of the Friedman test was 8.8089^{-15}, suggesting the existence of significant differences among the tested algorithms. Generally, HOA-1 outperformed other algorithms in the benchmark functions tests. Compared with HOA-1, HOA-2 performed better in F15 than the others. However, it should be noted that HOA-2 is also an effective algorithm, because the mean rank of HOA-2 was fifth among the nine algorithms according to the Friedman test result in Table 5. The reason HOA-1 performs better than HOA-2 in benchmark functions test can be explained by two factors. The first reason is the small number of I_s which makes it difficult to fully embody the advantages of Gaussian distribution multiple dimensions sampling in HOA-2. The second reason is that the landscapes of benchmark functions are simpler than the actual engineering problems. Therefore, a one-dimensional position update used in HOA-1 can efficiently search the space.

The history of minimum-value searching of benchmark functions is diagrammed in Figure 4. The gentle decrease in the fitness value means the exploitation search was effectively conducted, while the sharp or even vertical decline reflects a better result found by the exploration search. From this point of view, the exploration abilities of GA and HHO are outstanding, and the exploitation ability of HOA-1 is conspicuous. The stepped attenuation of search step length of S-particles attributes more to the descent slope changes in HOA-1 and HOA-2. Note that the number of function evaluations, which is equal to the number of individuals generated in the search space, is used as the coordinate of x-axis. Accordingly, the coordinates of the x-axis of the final minimum values of HOA-1 and HOA-2 are not the same as other compared algorithms, due to the smaller function evaluations number, which is amplified and shown in the subplot of F1 function in Figure 4.

To dissect the search process and convergence of two proposed algorithms, we selected F1, F8 and F14 to represent three types of functions with which to analyze the characteristics of iterative search. The fitness history curves of solving three functions by HOA-1 and HOA-2 are delineated in Figure 5. The fitness history curves show that B-particles in HOA-1 converge faster than those in HOA-2, while B-particles in HOA-2 have better potential for local minima avoidance, which is reflected by the intermittent sharp decline during the iterative search in Figure 5b,c. The initial and final positions of all B-particles of the search in HOA-1 and HOA-2 are displayed in Figures 6 and 7, respectively. Note that the positions of B-particles are represented by the first two coordinates. The B-particles gathered in a subdomain of the search space after the iterative search, which demonstrated that both algorithms possess the ability to drive B-particles to converge and find a high-precision solution.

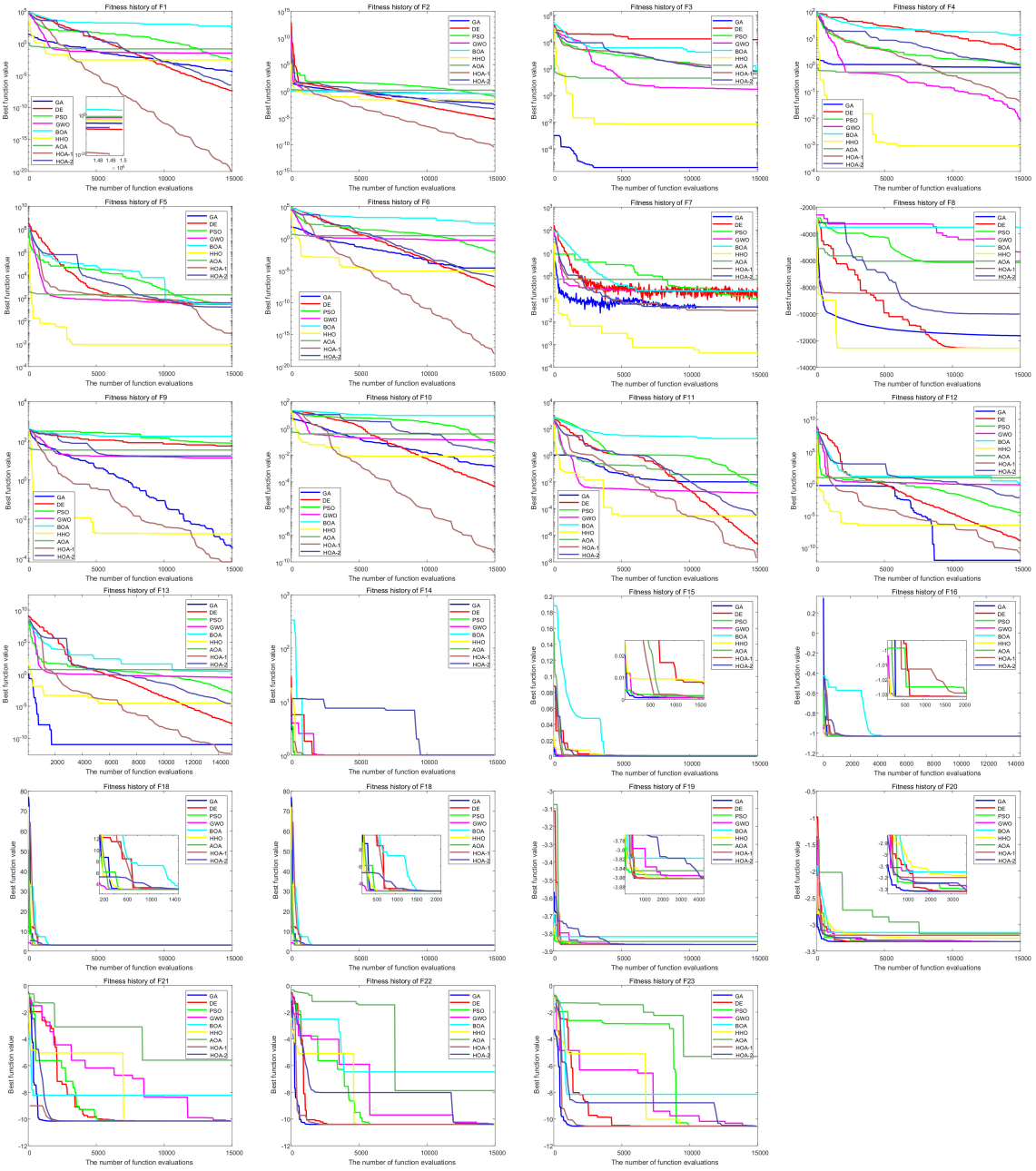
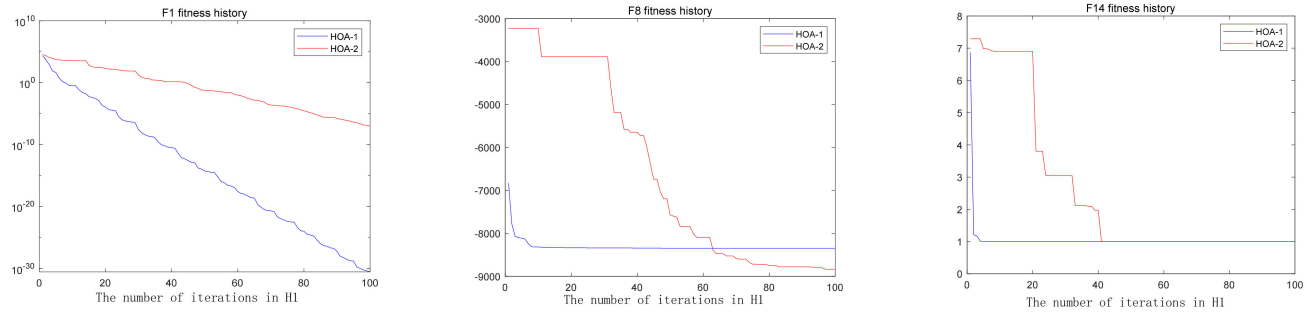
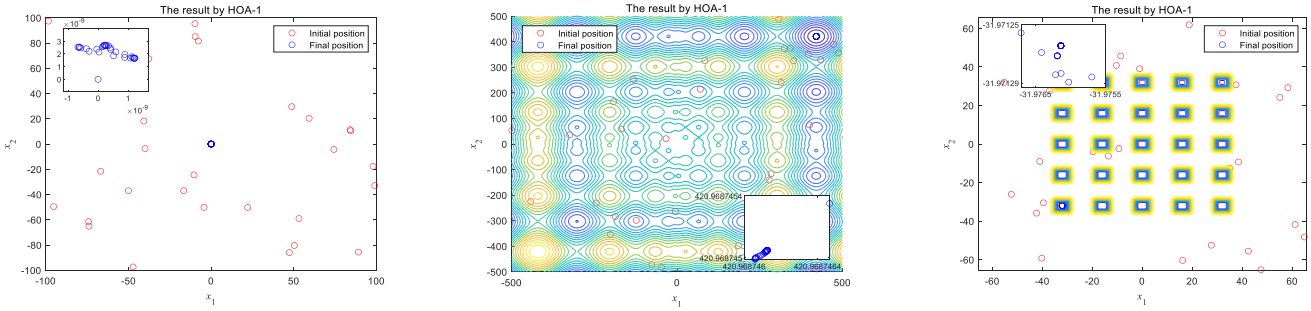


Figure 4. Searching history of benchmark functions.



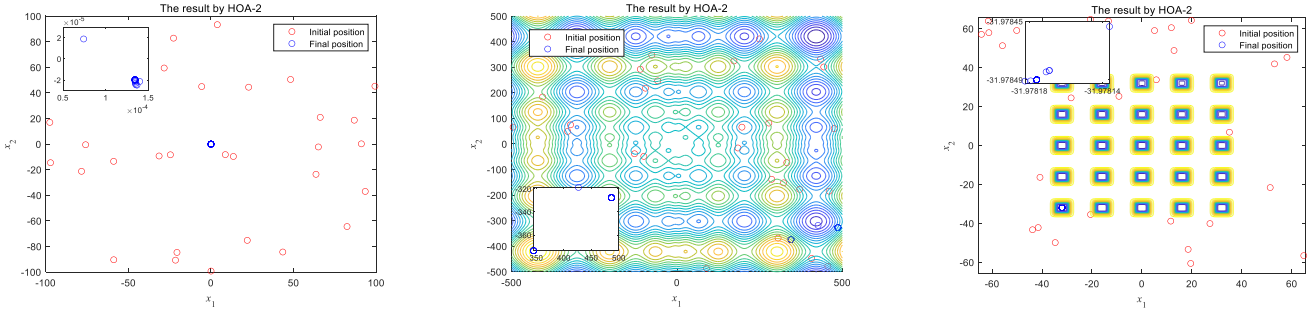
(**a**) (**b**) (**c**)

Figure 5. The history fitness curves solved by HOA-1 and HOA-2 for (**a**) F1; (**b**) F8; (**c**) F14.



(**a**) (**b**) (**c**)

Figure 6. Initial and final positions of B-particles solved by HOA-1 for (**a**) F1; (**b**) F8; (**c**) F14.



(**a**) (**b**) (**c**)

Figure 7. Initial and final positions of B-particles solved by HOA-2 for (**a**) F1; (**b**) F8; (**c**) F14.

Table 2. Results of unimodal benchmark functions.

F		HOA-1		HOA-2		GA	DE	PSO	GWO	BOA	HHO	AOA
F1	*B*	3.28×10^{-16}	$T = 6$	3.16×10^{-7}	$T = 15$	6.73×10^{-9}	1.12×10^{-8}	1.13×10^{-3}	1.88×10^{-6}	68.0	2.43×10^{-6}	0.134
	A	8.87×10^{-16}		9.34×10^{-6}		1.33×10^{-8}	1.13×10^{-2}	0.0157	3.18×10^{-2}	407	0.0106	0.172
	S	3.28×10^{-16}		8.63×10^{-6}		3.78×10^{-9}	0.0542	0.0127	0.0153	155	0.0241	0.0157
F2	*B*	4.38×10^{-10}	$T = 8$	1.05×10^{-4}	$T = 15$	1.92×10^{-4}	4.92×10^{-6}	0.0568	0.174	0.225	9.46×10^{-3}	1.65
	A	0.237		5.21×10^{-4}		3.74×10^{-4}	8.67×10^{-6}	0.175	0.417	2.39	0.0489	1.81
	S	1.10		6.26×10^{-3}		9.17×10^{-5}	2.57×10^{-6}	0.819	0.137	1.67	0.0368	0.0763
F3	*B*	75.7	$T = 40$	35.2	$T = 30$	1.14×10^{-18}	1.54×10^{4}	40.8	2.27	72.9	3.62×10^{-4}	14.1
	A	314		189		7.04×10^{-11}	1.88×10^{4}	99.7	6.93	497	0.142	73.1
	S	146		89.5		2.75×10^{-10}	2.64×10^{3}	35.9	2.74	325	0.231	107
F4	*B*	0.0302	$T = 35$	0.664	$T = 30$	0.804	3.59	0.931	5.38×10^{-3}	12.2	1.52×10^{-5}	0.500
	A	0.136		2.39		1.76	6.26	1.35	0.274	16.0	4.94×10^{-3}	0.504
	S	0.191		1.24		0.547	3.15	0.240	0.212	1.33	3.78×10^{-3}	6.63×10^{-3}
F5	*B*	0.173	$T = 40$	14.7	$T = 30$	N/A	21.8	33.3	36	29.0	5.76×10^{-4}	183
	A	52.6		41.1		N/A	112	190	64.1	29.1	0.0186	223
	S	33.2		23.8		N/A	81.8	223	14.6	0.158	0.0256	15.0
F6	*B*	1.36×10^{-18}	$T = 10$	3.35×10^{-7}	$T = 15$	1.55×10^{-5}	5.82×10^{-9}	3.59×10^{-3}	5.80×10^{-5}	140	3.37×10^{-8}	2.57
	A	2.82×10^{-18}		3.51×10^{-5}		4.93×10^{-5}	6.82×10^{-4}	0.0201	0.718	465	9.17×10^{-5}	3.20
	S	1.03×10^{-18}		5.63×10^{-5}		3.46×10^{-5}	3.67×10^{-3}	0.0217	0.365	162	1.29×10^{-4}	0.286
F7	*B*	0.0135	$T = 70$	0.0329	$T = 25$	0.0303	0.0818	0.168	0.0923	0.204	5.99×10^{-5}	0.684
	A	0.0480		0.0747		0.0916	0.198	0.377	0.191	0.299	9.79×10^{-4}	0.724
	S	0.0192		0.0228		0.0364	0.0572	0.147	0.0500	0.0691	1.11×10^{-3}	0.0133

Table 3. Results of multimodal benchmark functions.

F		HOA-1		HOA-2		GA	DE	PSO	GWO	BOA	HHO	AOA
F8	*B*	-9.19×10^3	$T = 40$	-10.0×10^4	$T = 25$	-1.21×10^4	-1.26×10^4	-8.23×10^3	-7.51×10^3	-3.53×10^3	-1.26×10^4	-6.21×10^3
	A	-7.40×10^3		-7.83×10^3		-1.13×10^4	-1.24×10^4	-5.56×10^3	-5.95×10^3	-2.67×10^3	-1.26×10^4	-5.44×10^3
	S	604		924		399	142	1.53×10^3	924	372	0.941	405
F9	*B*	1.99×10^{-6}	$T = 40$	10.9	$T = 10$	6.51×10^{-5}	53.7	39.3	5.77	177	4.19×10^{-5}	33.9
	A	1.02×10^{-4}		39.0		2.34×10^{-1}	65.0	68.5	19.3	210	0.0219	37.3
	S	3.44×10^{-5}		18.1		0.493	7.29	16.7	8.04	15.5	0.0238	1.46
F10	*B*	4.21×10^{-7}	$T = 7$	0.0255	$T = 30$	1.05×10^{-3}	3.06×10^{-5}	0.0380	1.36×10^{-3}	9.13	2.90×10^{-4}	0.379
	A	0.211		1.54		2.73×10^{-3}	0.0209	0.473	0.175	10.5	7.91×10^{-3}	0.435
	S	0.636		0.772		1.09×10^{-3}	0.109	0.530	0.0738	0.557	6.15×10^{-3}	0.0322
F11	*B*	2.60×10^{-8}	$T = 20$	2.52×10^{-7}	$T = 15$	6.08×10^{-6}	3.91×10^{-8}	1.64×10^{-4}	7.66×10^{-4}	13.0	6.18×10^{-6}	0.0324
	A	0.0339		0.0106		3.81×10^{-3}	1.02×10^{-3}	0.0107	6.87×10^{-3}	19.3	1.51×10^{-3}	0.253
	S	0.0276		0.0123		2.08×10^{-3}	2.69×10^{-3}	9.58×10^{-3}	9.10×10^{-3}	2.50	2.60×10^{-3}	0.196
F12	*B*	6.87×10^{-12}	$T = 20$	1.71×10^{-6}	$T = 30$	3.47×10^{-11}	5.44×10^{-10}	2.93×10^{-5}	0.667	1.01	5.85×10^{-8}	7.33
	A	4.15×10^{-11}		2.18		3.84	3.07×10^{-3}	3.68×10^{-3}	1.36	5.08	2.48×10^{-5}	8.58
	S	6.81×10^{-11}		1.44		6.03	8.78×10^{-3}	0.0186	0.408	5.27	3.43×10^{-5}	0.412
F13	*B*	8.78×10^{-14}	$T = 15$	1.92×10^{-6}	$T = 20$	3.41×10^{-13}	3.27×10^{-9}	5.68×10^{-4}	0.346	2.85	2.21×10^{-6}	4.76
	A	3.67×10^{-13}		0.0505		0.0549	0.151	9.20×10^{-3}	0.999	3.03	3.92×10^{-4}	5.27
	S	2.04×10^{-13}		0.137		0.236	0.523	7.49×10^{-3}	0.367	0.0993	3.41×10^{-4}	0.192

Table 4. Results of fixed-dimension multimodal benchmark functions.

F		HOA-1		HOA-2		GA	DE	PSO	GWO	BOA	HHO	AOA
F14	*B*	0.998	$T = 25$	0.998	$T = 25$	0.998	0.998	0.998	0.998	0.998	0.998	0.998
	A	8.04		2.71		7.51	1.16	3.13	3.58	3.13	1.097	6.47
	S	0.0139		1.87		4.30	0.885	2.43	3.32	2.35	0.298	1.64
F15	*B*	3.09×10^{-4}	$T = 20$	3.07×10^{-4}	$T = 25$	5.55×10^{-4}	4.97×10^{-4}	3.70×10^{-4}	3.60×10^{-4}	8.53×10^{-4}	3.08×10^{-4}	8.56×10^{-4}
	A	8.48×10^{-3}		2.01×10^{-3}		2.85×10^{-3}	7.65×10^{-4}	8.98×10^{-4}	5.61×10^{-3}	0.0173	4.03×10^{-4}	0.0199
	S	0.0139		4.92×10^{-3}		5.08×10^{-3}	2.03×10^{-4}	2.64×10^{-4}	7.04×10^{-3}	0.0270	2.21×10^{-4}	0.0207
F16	*B*	−1.03	$T = 20$	−1.03	$T = 30$	−1.03	−1.03	−1.03	−1.03	−1.03	−1.03	−1.03
	A	−1.03		−1.03		−1.03	−1.03	−1.03	−1.03	−1.02	−1.03	−1.03
	S	2.57×10^{-12}		1.22×10^{-5}		2.14×10^{-9}	0	0	7.11×10^{-8}	0.0120	1.24×10^{-7}	1.56×10^{-7}
F17	*B*	−0.398	$T = 20$	0.398	$T = 30$	0.398	0.398	0.398	0.398	0.398	0.398	0.398
	A	−0.398		0.398		0.398	0.398	0.398	0.398	0.937	0.398	0.411
	S	1.83×10^{-12}		1.89×10^{-8}		1.65×10^{-9}	2.66×10^{-15}	1.11×10^{-16}	1.50×10^{-4}	1.41	4.84×10^{-5}	7.51×10^{-3}
F18	*B*	3.00	$T = 20$	3.00	$T = 40$	3.00	3.00	3.00	3.00	3.02	3.00	3.00
	A	3.00		3.00		3.00	3.90	3.00	5.70	10.7	3.00	23.3
	S	9.42×10^{-11}		1.78×10^{-5}		1.60×10^{-7}	4.85	4.40×10^{-15}	14.5	7.62	2.06×10^{-8}	21.1
F19	*B*	−3.86	$T = 20$	−3.86	$T = 40$	−3.86	−3.86	−3.86	−3.86	−3.84	−3.86	−3.86
	A	−3.86		−3.86		−3.86	−3.86	−3.86	−3.86	−3.53	−3.86	−3.84
	S	4.83×10^{-13}		9.00×10^{-7}		3.48×10^{-8}	2.66×10^{-15}	3.00×10^{-3}	5.43×10^{-3}	0.288	6.04×10^{-3}	8.22×10^{-3}
F20	*B*	−3.32	$T = 30$	−3.32	$T = 40$	−3.32	−3.32	−3.32	−3.32	−3.09	−3.26	−3.21
	A	−3.28		−3.30		−3.28	−3.32	−3.29	−3.24	−2.27	−3.07	−3.04
	S	0.0573		0.0476		0.0573	1.68×10^{-6}	0.0503	0.119	0.467	0.0997	0.119
F21	*B*	−10.1532	$T = 80$	−10.1489	$T = 80$	−10.1532	−10.1532	−10.1532	−10.1529	−6.11	−10.1508	−5.91
	A	−6.31		−7.20		−5.88	−8.84	−6.89	−9.646	−2.83	−8.4191	−3.20
	S	3.46		3.41		2.97	2.43	3.37	1.516	1.49	2.2256	0.850
F22	*B*	−10.4029	$T = 80$	−10.3931	$T = 80$	−10.4029	−10.4029	−10.4029	−10.4028	−8.55	−10.4028	−7.87
	A	−5.98		−7.29		−5.76	−10.002	−9.52	−10.225	−2.91	−8.7108	−3.45
	S	3.47		3.55		2.91	1.096	2.23	0.947	1.49	2.4287	0.971
F23	*B*	−10.5364	$T = 80$	−10.5318	$T = 80$	−10.5364	−10.5364	−10.5364	−10.5357	−6.97	−10.5358	−9.93
	A	−6.05		−7.37		−5.41	−10.428	−8.99	−10.354	−3.28	−8.3461	−4.42
	S	3.74		3.81		3.46	0.379	2.81	0.970	1.69	2.4791	2.44

Table 5. Ranking-based Friedman test for the comparative algorithms.

F	HOA-1	HOA-2	GA	DE	PSO	GWO	BOA	HHO	AOA
F1	1	4	2	3	7	5	9	6	8
F2	1	3	4	2	6	7	8	5	9
F3	8	5	1	9	6	3	7	2	4
F4	3	5	6	8	7	2	9	1	4
F5	2	3	9	4	6	7	5	1	8
F6	1	4	5	2	7	6	9	3	8
F7	2	4	3	5	7	6	8	1	9
F8	5	4	3	1.5	6	7	9	1.5	8
F9	1	6	3	7	8	4	9	2	5
F10	1	6	4	2	7	5	9	3	8
F11	1	3	4	2	6	7	9	5	8
F12	1	5	2	3	6	7	8	4	9
F13	1	4	2	3	6	7	8	5	9

Table 5. *Cont.*

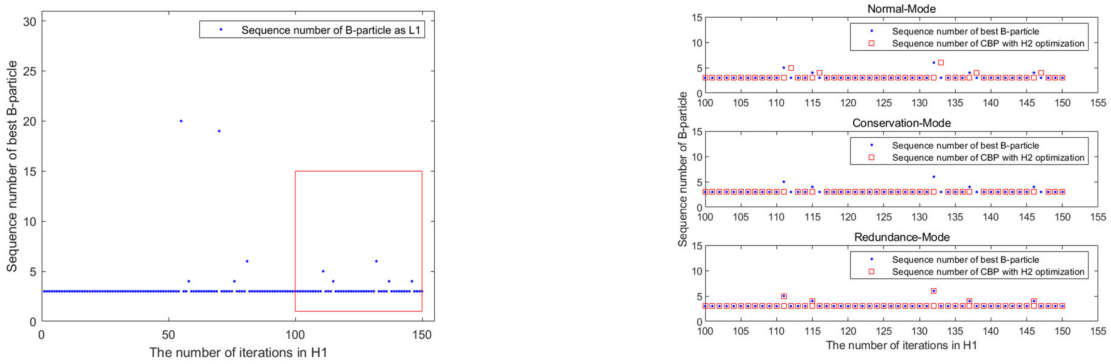
F	HOA-1	HOA-2	GA	DE	PSO	GWO	BOA	HHO	AOA
F14	5	5	5	5	5	5	5	5	5
F15	3	1	7	6	5	4	8	2	9
F16	5	5	5	5	5	5	5	5	5
F17	5	5	5	5	5	5	5	5	5
F18	4.5	4.5	4.5	4.5	4.5	4.5	9	4.5	4.5
F19	4.5	4.5	4.5	4.5	4.5	4.5	9	4.5	4.5
F20	3.5	3.5	3.5	3.5	3.5	3.5	9	7	8
F21	2.5	7	2.5	2.5	2.5	5	8	6	9
F22	2.5	7	2.5	2.5	2.5	5.5	8	5.5	9
F23	2.5	7	2.5	2.5	2.5	6	9	5	8
Mean Rank	2.869565	4.586957	3.913043	4.021739	5.434783	5.26087	7.913043	3.869565	7.130435

4.2.2. The Mechanism Explanation by Examples

The core idea of HOF is based on a competition mechanism that tilts computing resources to CBP to make it better, and compels all B-particles to compete for the rewards assigned to the CBP. There is one point to be emphasized: in both HOA-1 and HOA-2, the reward for CBP is granted before the corresponding H1 iteration starts. In other words, once a B-particle takes the title of CBP after one H1 iteration, the reward is conferred on it regardless of whether the alternation of CBP appears in the next iteration. This strategy guarantees the execution of one H2 optimization in every H1 iteration, and is named Normal-Mode, which implies that the number of H2 optimization implementation may not be fixed to 1. There are another two possible situations. One case, called Conservation-Mode, is when H2 optimization is implemented when the *i*th B-particle becomes CBP and keeps the place until its next personal H1 iteration comes. If other B-particles assume the place of CBP in this period, the *i*th B-particle will not gain extra assistance from S-particles. The other case is defined as Redundance-Mode, in which H2 optimization is executed as soon as the *i*th B particle becomes CBP. In this case, there may be more than one H2 optimization in one H1 iteration. These three situations reveal the time delay between the reward of CBP winning and awarding. From the perspective of computational stability, we embed Normal-Mode in HOF to create HOA-1 and HOA-2. By the phenomenon above, this competition mechanism strongly stimulates the competition in B-particles, especially when the numbers of iterations and S-particles in H2 is increased, which exacerbates the imbalance between the CBP and other B-particles.

To make a specific impression on readers, simulated examples were carried out to provide more concrete specification. For better demonstration of the competition mechanism, we used HOA-1 to find the minimum of F8 function and set I_b to 150 and I_s to 3 in order to weaken the preponderance of CBP. To show the competition for the CBP place among B-particles, we adjusted the parameter T to 10 to preferably exhibit the stepped attenuation of search range.

The situation of CBP alternation is shown in Figure 8, in which Figure 8a gives the alternation of CBP in each H1 iteration and Figure 8b presents the actual execution of H2 optimization in the local region of the red square in Figure 8a. The differences between three situations can be easily observed: in Normal-Mode the H2 optimization is carried out in every H1 iteration, while the number of H2 optimization in Conservation-Mode and in Redundance-Mode decreases five times and increases five times, respectively (the number is 10 in the whole process). Note that in this example, at most, two B-particles become the CBP in one H1 iteration. If the competition is fiercer and more B-particles sequentially become the CBP in one H1 iteration, another example will reflect the result in Figure 9, in which the numbers of H2 optimization carried out in three situations are respectively 16, 14 and 20. In Conservation-Mode, H2 optimization is implemented only when one B-particle remains as CBP for at least two successive H1 iterations. For the performance comparison of different algorithms, Normal-Mode is preferred, due to its fixed amount of computation.



(**a**) (**b**)

Figure 8. Alternation of L1 and the comparison of three modes (**a**) sequence number of the best B-particle in H1 iteration; (**b**) Local diagram of the sequence number of B-particle with H2 optimization in three modes.

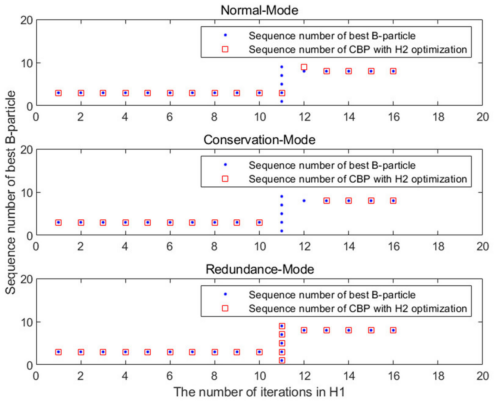


Figure 9. Complicated example for three modes comparison.

Another typical mechanism is the stepped attenuation of $deltaS^n$, which is shown in Figure 10. Because the upper bound and lower bound of each dimension of F8 function is the same, the first element of $deltaS^n$ is qualified to embody its range variation. All elements of $deltaS^n$ are reduced to one tenth every 10 H1 iterations. However, it was found in the experiment that the algorithm may not converge to the global optima if the value of T is too small, while a T that is too large causes a loss of solution accuracy in a finite number of iterations. Therefore, further research is warranted to resolve this contradiction.

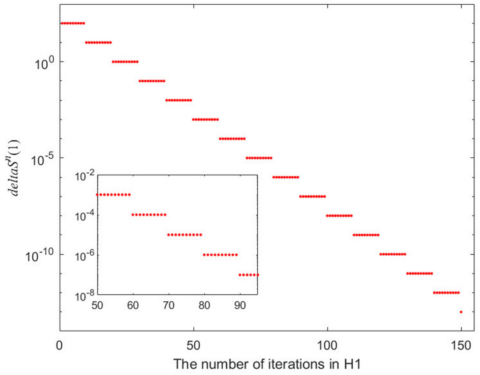


Figure 10. Stepped decrease of $deltaS^n$.

4.2.3. The Value of Parameter T

In the benchmark functions test, a significant feature of both HOA-1 and HOA-2 is that the parameter T is usually different and tuned according to the test function, which is

due to the shrinking mechanism of search space. When the number of iterations in H1 is fixed, decreasing the value of T means to accelerate the process of space shrinking. If this rate is too fast, the minimum search by S-particles will be severely restricted and limited in its ability to help CBP update its position. Only when the optimization in H2 works properly can HOA-1 and HOA-2 seek out a good solution. Conversely, a slow shrinking rate reduces the search efficiency, due to the search space being too large. Meanwhile, a finite iterative number inevitably limits the amount of computation. Hence, in order to promote the smooth progress of optimization, we used different constant values to set the parameter T to control the search precision.

For better elucidation of the influence of different T on a given function, an additional example was provided. Without loss of generality, the F1 function was used to present the phenomenon of performance deterioration and the HOA-1 was assigned to execute the optimization mission. In HOA-1, we added I_s from 4 to 10 and set the value of T as 4, 5, 6 and 12. These four group results were collected and compared to show the difference. Each group had 30 solutions, ranked in descending order of F1 function, and all results are displayed in Figure 11, in which Fbest represents the minimum after each run, and the common logarithm of Fbest is taken to conveniently reflect the difference. Three key features can be concluded:

1. A smaller T has the potential to make the algorithm find a better solution.
2. A destabilization of the search may occur when the value of T varies.
3. A T that is too small may impede the algorithm search for the optimal solution.

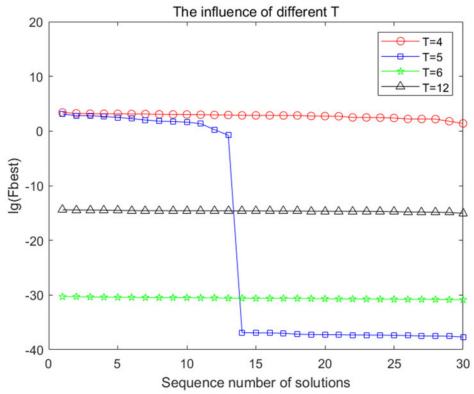


Figure 11. The influence of different values of T on optimized results.

In this work, the value of T was set by trial and error to achieve a balance between stability and search efficacy, which leaves room for further research to design an adaptive law for parameter adjustment and to mitigate the trouble of manual parameter setting.

5. Applications of the Proposed Methods

In this section, two spacecraft trajectory optimization problems, the multi-impulse minimum fuel orbit transfer and the pursuit-evasion game of two spacecraft, are employed to reflect the capability of the proposed HOA-1 and HOA-2. The former problem is extensively solved by meta-heuristic algorithms [33,44]. The latter problem considers the requirement of tracking a non-cooperative target with maneuvering ability, and has been the subject of much recent attention [45]. The functions of both problems demonstrate strong nonlinearity and variable coupling, even without additional constraints.

5.1. Multi-Impulse Minimum Fuel Orbit Transfer

5.1.1. Problem Formulation and Parameter Setting

Spacecraft orbit optimal transfer by impulsive thrusters is a classical and significant problem. However, an analytic solution of optimal impulsive control for orbit transfer in complicated situations is difficult to obtain when the number of impulses increases

and additional complicated constraints are considered. Therefore, a direct approach was developed to construct a general nonlinear programming problem and solve it by advanced optimization algorithms [46]. Relevant research results can be referred to in [44].

Considering the goal of proving the effectiveness of the proposed methods, we chose minimum fuel transfer between two coplanar circular orbits as the problem to be optimized. This has an analytical solution known as the famous Hohmann transfer with two-impulse maneuver. Usually, the fuel consumed is positively correlated with the velocity increment, so the orbit transfer by minimum velocity increment is equivalent to the minimum fuel case. For additional details on the derivation, [47] is recommended. In accordance with the size limit of this paper, a brief introduction of Hohmann transfer is generalized below.

For two coplanar circular orbits around the celestial body, an ellipse connecting two orbits and tangent to them is shown in Figure 12. A spacecraft on the initial orbit can reach the final orbit after two-impulse maneuver $\Delta v1$ and $\Delta v2$. The transfer trajectory is half an ellipse from the perigee to the apogee of the transfer elliptic orbit. Two impulses $\Delta v1$ and $\Delta v2$ are given as follows:

$$\Delta v1 = \sqrt{2\mu \frac{r2}{r1(r1+r2)}} - \sqrt{\frac{\mu}{r1}}, \ \Delta v2 = \sqrt{\frac{\mu}{r2}} - \sqrt{2\mu \frac{r1}{r2(r1+r2)}} \tag{13}$$

where μ is the gravitational constant of the celestial body, $r1$ and $r2$ are radii of the initial and final circular orbit. According to the characteristic of Hohmann transfer with the theoretical solution of two-impulse maneuver, we designed three cases to test the optimization performance of the proposed two algorithms. As with problem 1 in [44], they concerned two orbits around Mars ($\mu = 42,830$ km/s^2), with $r1$ and $r2$ being 8000 km and 15,000 km, respectively. The three cases are distinguished by search space as listed in Table 6.

Table 6. Search space of three cases.

Case	Search Space
Case 1	2-dimension, $Lb = [-0.1\ -0.1]^T$ and $Ub = [0.8\ 0.8]^T$
Case 2	5-dimension, $Lb = [0\ 0\ 5577\ 0\ 0]^T$ and $Ub = [0.25\ 0.25\ 33{,}465\ 0.25\ 0.25]^T$
Case 3	5-dimension, $Lb = [0\ 0\ 5577\ 0\ 0]^T$ and $Ub = [0.25\ 0.25\ 16{,}733\ 0.25\ 0.25]^T$

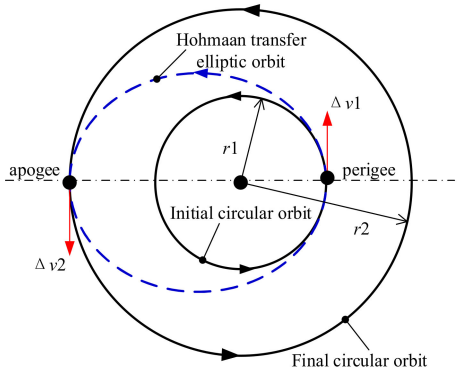


Figure 12. Diagram of Hohmann transfer.

Without loss of generality, the start points of the three cases were all set to the (8000,0). In Case 1, we gave the first in-plane impulse to make the spacecraft leave the initial orbit and applied the second impulse to make it revolve on the final orbit when it reached one point of the final orbit. When the first impulse was provided, the second impulse, if the transfer trajectory intersected the final orbit, was able to be calculated. Therefore, the search space was a 2-dimensional vector whose components were the radial and tangential velocity increments. The landscape is depicted in Figure 13, in which Δv_x and Δv_y represent coordinate components of Δv_1, and the domain of the initial velocity impulse that cannot

realize orbit transfer is marked with a red dot. Subsequently, we shrank the range of initial velocity increments so as to make the transfer impossible by two-impulse, by which a three-impulse orbit transfer was constructed and the search space became a five-dimensional vector, added to the second 2-dimension impulse and the applied time. Different values of the upper bound of the applied time led to Case 2 and Case 3. In Case 2, the upper bound of applied time allowed the spacecraft to fly one revolution and return to the start point, by which the same total impulse increment with two-impulse Hohmann transfer could be achieved by three-impulse maneuver. However, the upper bound of applied time violated this condition in Case 3, making the result of three-impulse orbit transfer by minimum velocity increment different from Case 2.

DE, GA, PSO, GWO, HHO, AOA, HOA-1 and HOA-2 were used to solve the three cases according to the result of benchmark functions test. The numbers of iterations Imax of all these contrastive algorithms were all set to 400, and the population size was 50. Considering the inconsistency between the dimension of search space and the number of search individuals of these algorithms, we set both I_b and I_s to 30. The number of S-particles in HOA-1 is equal to N, while that can be changed to $4N$ in HOA-2 to ensure that the total number of function evaluations is the same as the comparison algorithms. The parameters of DE, GA, PSO, GWO, HHO and AOA were the same as those in Table 1. For more meticulous search, we set α to 0.1 and c to 0 in both HOA-1 and HOA-2.

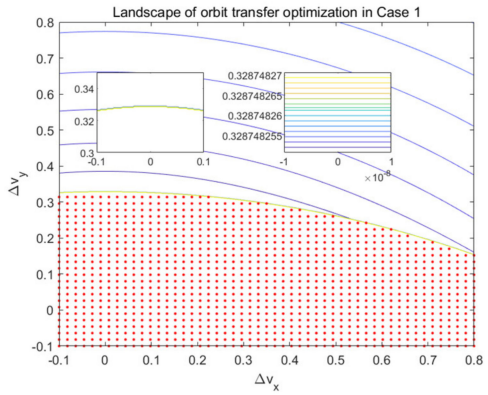


Figure 13. The landscape of orbit transfer in Case 1.

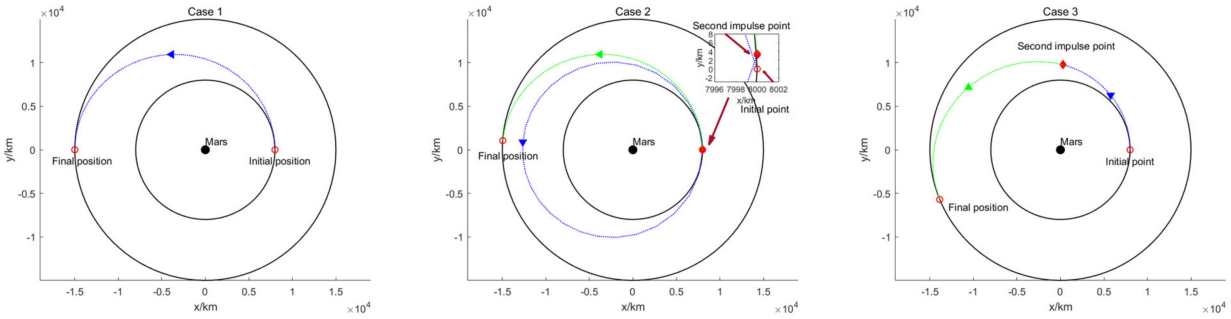
5.1.2. Results and Discussion

For Case 1 and Case 2, adding in $\Delta v1$ and $\Delta v2$ according to Equations (13), the theoretical minimum velocity increment is 0.6091531 km/s^2. Each algorithm was run 30 times and the best result have been selected and listed in Table 7, in which the best solutions found by these algorithms are bold. The results show that HOA-2 found the best solutions for the three cases, which evidently reflects the superiority of the proposed methods. Because the number of S-particles of HOA-1 should be equal to the problem dimension N, and that of HOA-2 can be properly set to $4N$, the amount of computation of HOA-1 is noticeably less than other algorithms, which explains the performance degradation of HOA-1.

Table 7. Results of three cases.

Algorithms	Case 1	Case 2	Case 3
DE	0.609153269	0.609856799	0.616650300
GA	0.609154556	0.609274226	0.634113176
PSO	0.609155127	0.609163945	0.613222985
HHO	0.609158146	0.609227931	0.613319255
AOA	0.609333116	0.609173849	0.613235349
GWO	0.609156659	0.609232163	0.612584848
HOA-1	0.609156393 (T = 10)	0.609189193 (T = 15)	0.613940295 (T = 15)
HOA-2	0.609153802 (T = 10)	0.609153512 (T = 15)	0.612572700 (T = 15)

The process of orbit transfer using the best solution of three cases is diagrammed in Figure 14, which is consistent with the previous analysis and proves the correctness of the results. Comparing the results of Case 1 and Case 2, it can be observed that the difficulty of finding the optimal solution increases when the dimensions of the search space increase. In a similar way, further reduction in velocity range can construct the orbit transfer by more impulses and increase the difficulty of spacecraft orbit maneuver optimization. Nevertheless, uncertain spacecraft trajectory is more likely with the increasing number of impulses [48].



(**a**) (**b**) (**c**)

Figure 14. The orbit transfer of (**a**) Case 1; (**b**) Case 2; (**c**) Case 3.

5.2. Pursuit-Evasion Game of Two Spacecraft

5.2.1. Problem Formulation and Parameters Setting

Spacecraft pursuit-evasion is a typical game problem, in which the pursuing spacecraft attempts to capture the target while the evading spacecraft strives for escape. Accordingly, this pursuit-evasion problem is a zero-sum differential game. As a result of the great difficulty of an analytical solution derivation, the numerical methods have been extensively used in recent decades. The indirect method of solving this problem is to derive the necessary optimal conditions, transform it into a two-point boundary value problem (TPBVP) and find a saddle point by numerical methods. In this paper, the TPBVP solution is used to test the performance of the proposed methods. The dynamic model and necessary conditions derivation are adopted from [49], and a transcription is provided below.

The differential equations of the pursuit-evasion game are written as follows:

$$\dot{X} = A(t)X + T_P U_P + T_E U_E \tag{14}$$

where $X = [x_P, y_P, z_P, \dot{x}_P, \dot{y}_P, \dot{z}_P, x_E, y_E, z_E, \dot{x}_E, \dot{y}_E, \dot{z}_E]^T$ is the state variable of positions and velocities of two spacecraft in the LVLH coordinate system of a virtual spacecraft in circular reference orbit, and the subscripts P and E represent the pursuing spacecraft and the evading spacecraft, respectively. $U_P = [0_{1\times3}, u_{Px}, u_{Py}, u_{Pz}, 0_{1\times6}]^T$ and $U_E = [0_{1\times9}, u_{Ex}, u_{Ey}, u_{Ez}]^T$ represent the direction vector of control exerted on two spacecraft, and satisfy the restraint conditions $\sqrt{u_{Ix}^2 + u_{Iy}^2 + u_{Iz}^2} \leq 1$, in which I means P or E. T_P and T_E are the value of maximum thrust per unit mass of two spacecraft. Clohessy–Wiltshire equations [50] are used to describe the relative motion of two spacecraft with respect to the reference rotating coordinate system, and the matrix $A(t)$ can be expressed by Equation (15).

$$A(t) = \begin{bmatrix} A_P(t) & 0 \\ 0 & A_E(t) \end{bmatrix}, \; A_P(t) = A_E(t) = \begin{bmatrix} 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 \\ 3\omega^2 & 0 & 0 & 0 & 2\omega & 0 \\ 0 & 0 & 0 & -2\omega & 0 & 0 \\ 0 & 0 & -\omega^2 & 0 & 0 & 0 \end{bmatrix} \tag{15}$$

where ω is the constant rotational angular velocity of reference coordinate system in the gravitational field of the Earth. Set the terminal time as t_f, so the payoff function is as follows:

$$J = \frac{1}{2}[(x_P - x_E)^2 + (y_P - y_E)^2 + (z_P - z_E)^2]\Big|_{t=t_f} \tag{16}$$

Therefore, the spacecraft pursuit-evasion game can be described thus: the goal of pursuing spacecraft is to change U_P to minimize J, while the evading spacecraft aims at maximum by J by U_E.

The adjoint variable $\lambda = [\lambda_P, \lambda_E]^T$ is introduced corresponding to X, in which $\lambda_I = [\lambda_{Ix}, \lambda_{Iy}, \lambda_{Iz}, \lambda_{I\dot{x}}, \lambda_{I\dot{y}}, \lambda_{I\dot{z}}]^T$ and $I = P, E$. Construct the Hamiltonian function as $H = \lambda^T X$, and the control of two spacecraft can be derived according to the minimum principle and Cauchy–Schwarz inequality.

$$\begin{cases} u_{Px} = -\frac{\lambda_{Px}}{\sqrt{\lambda_{Px}^2+\lambda_{Py}^2+\lambda_{Pz}^2}} \\ u_{Py} = -\frac{\lambda_{Py}}{\sqrt{\lambda_{Px}^2+\lambda_{Py}^2+\lambda_{Pz}^2}} \\ u_{Pz} = -\frac{\lambda_{Pz}}{\sqrt{\lambda_{Px}^2+\lambda_{Py}^2+\lambda_{Pz}^2}} \end{cases}, \begin{cases} u_{Ex} = \frac{\lambda_{Ex}}{\sqrt{\lambda_{Ex}^2+\lambda_{Ey}^2+\lambda_{Ez}^2}} \\ u_{Ey} = \frac{\lambda_{Ey}}{\sqrt{\lambda_{Ex}^2+\lambda_{Ey}^2+\lambda_{Ez}^2}} \\ u_{Ez} = \frac{\lambda_{Ez}}{\sqrt{\lambda_{Ex}^2+\lambda_{Ey}^2+\lambda_{Ez}^2}} \end{cases} \tag{17}$$

The adjoint variable $\lambda = [\lambda_P, \lambda_E]^T$ satisfies the differential equations in (18).

$$\dot{\lambda} = -\left(\frac{\partial H}{\partial X}\right)^T \tag{18}$$

The transversal condition is derived by (19).

$$\lambda(t_f) = -\left(\frac{\partial J}{\partial X(t_f)}\right)^T = [\lambda_{Px}, \lambda_{Py}, \lambda_{Pz}, 0_{1\times3}, \lambda_{Ex}, \lambda_{Ey}, \lambda_{Ez}, 0_{1\times3}]^T_{t=t_f} \tag{19}$$

So far, the TPBVP has been derived completely. Input the guess of initial adjoint variables to the optimization algorithms and minimize J to test the performance of the proposed methods. Relevant parameters are set as follows:

$$\begin{cases} \omega = \sqrt{\mu/r_0^3},\ r_0 = 6900 \text{ km},\ \mu = 3.986 \times 10^5 \text{ km}^3/\text{s}^2 \\ t_f = 1500 \text{ s} \\ T_P = 0.02\ g,\ T_E = 0.01\ g,\ \ g = 0.0098 \text{ km/s}^2 \\ X(0) = [X_P(0), X_E(0)]^T \\ X_P(0) = [0, 5.12, 6.21, 0.0268, -4.715 \times 10^{-5}, 0.0011] \\ X_E(0) = [9.92, 24.12, 0, -0.2678, -0.005608, 0] \end{cases} \tag{20}$$

A reasonable initial value of adjoint variable λ_0 is given in (21), and the corresponding J is 0.489126783247148.

$$\begin{aligned} &\lambda_0 = [\lambda 1, \lambda 2, \ldots, \lambda 11, \lambda 12]^T \\ &\begin{cases} \lambda 1 = 1.828416833471 \times 10^{-3},\ \lambda 2 = -1.087432620919 \times 10^{-3} \\ \lambda 3 = -1.0996952525 \times 10^{-5},\ \lambda 4 = 1.31744718559753 \\ \lambda 5 = -0.092072105520415,\ \lambda 6 = 9.50453008156 \times 10^{-3} \\ \lambda 7 = -2.96673065336 \times 10^{-4},\ \lambda 8 = 1.01857391376 \times 10^{-4} \\ \lambda 9 = 1.4412237482 \times 10^{-5},\ \lambda 10 = -0.408955063275592 \\ \lambda 11 = 5.43535547235 \times 10^{-3},\ \lambda 12 = -8.00595379572 \times 10^{-4} \end{cases} \end{aligned} \tag{21}$$

According to λ_0, two cases were designed, as displayed in Table 8, to test the performance of the same eight algorithms as in Section 5.1. Note that in Table 8, $[1]_{12}$ is a 12-dimension column vector with all elements being 1. The parameters of these algorithms are almost the same as in Section 5.1, except that we changed I_b to 40 for HOA-1 to improve its performance, and the number of S-particles of HOA-1 and HOA-2 were respectively set to 12 and 20, to ensure a smaller amount of computation compared with other algorithms.

Table 8. Search space of two cases.

Case	Search Space
Case 1	$Lb = -5 \times [1]_{12}$, $Ub = 5 \times [1]_{12}$
Case 2	$Lb = \lambda_0 - 0.01 \times [1]_{12}$, $Ub = \lambda_0 + 0.01 \times [1]_{12}$

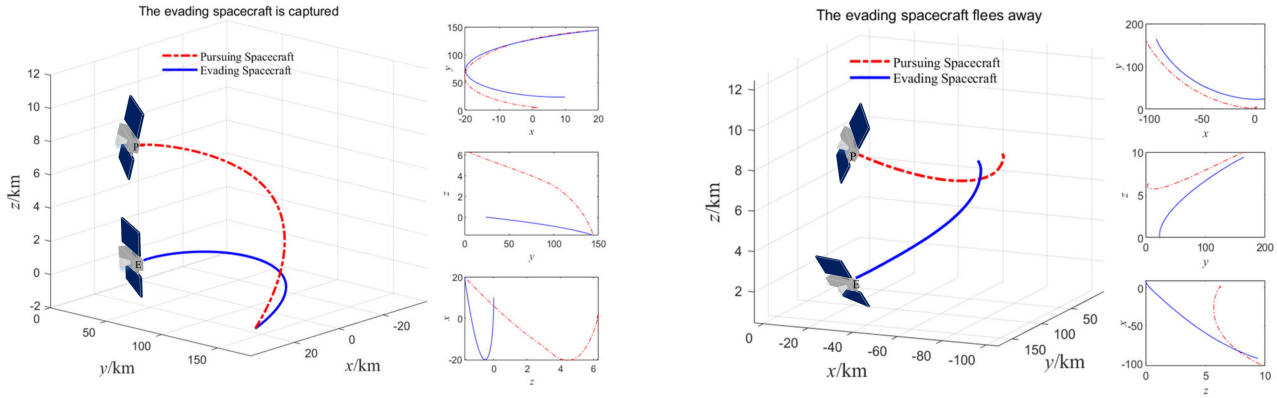
5.2.2. Results and Discussion

For both cases, each algorithm was run 10 times and the best result was selected and listed in Table 9, in which the best solutions are bold. The results show that HOA-2 performed best in this test, while the performance of HOA-1 is poorer than GWO and HOA-2, but better than other algorithms. Considering the difference between the two cases, it can be concluded that Case 1 examined the exploration of algorithms, while exploitation was more important in Case 2. Thus, the result is consistent with the previous statement that hierarchical optimization is designed to assign more computation resources to the CBP position update.

Table 9. Results of two cases.

Algorithms	Case 1	Case 2
DE	31.99873636	11.53267283
GA	38.30182027	7.804473121
PSO	9.975250323	13.99749289
GWO	9.945175394	1.523214099
HHO	27.19103058	2.48115544
AOA	26.93112607	10.05829695
HOA-1	11.56198501 (T = 4)	2.194148725 (T = 4)
HOA-2	7.148744461 (T = 4)	1.032140678 (T = 5)

It is universally accepted that the solution of TPBVP is difficult to acquire due to the strongly sensitive characteristic of the initial guess value of the adjoint variable that lacks physical meaning. However, a better initial guess may help us solve the TPBVP by the multiple shooting method, whose limitation lies in the high requirement of initial value. To illustrate the diversion caused by poorer initial adjoint variables, an example is provided in which two initial values of adjoint variable are used as input to the multiple shooting method to check whether the evading spacecraft could be captured by the designed control (17). The first initial value is λ_0, and the second one λ_0' is obtained by decreasing the first element of λ_0 by 20%. The payoff function J with λ_0' is 36.448. The result is diagrammed in Figure 15, which shows that a tiny change in the initial value of TPBVP had a great impact on the result. Therefore, the proposed methods can effectively help to guess the initial value of TPBVP.



(**a**) (**b**)

Figure 15. The result based on two sets of initial values of adjoint variables (**a**) the result with λ_0 as input; (**b**) the result with λ_0' as input.

In these two spacecraft trajectory optimization problems, HOA-2 performed better than HOA-1. This can be explained by two factors. The first is that the function evaluation number of HOA-1 is significantly less than other algorithms. The second factor is that the S-particle multidimensional sampling, based on the Gaussian distribution function in HOA-2, performs more efficiently when I_s increases. Therefore, the proper allocation of the numbers of B-particles and S-particles can make better use of hierarchical optimization algorithms. We suggest that HOA-1 is suitable when the problem dimension is relatively large, and otherwise HOA-2 is preferred, especially when the S-particles can be set to a much larger number than the problem dimension.

6. Conclusions

For improving the search ability of the best individual compared with others, a hierarchical optimization frame is proposed, in which B-particles synergistically explore the search space, and a local search performed by S-particles is conducted to update the position of the best B-particle. Based on this framework, two algorithms (HOA-1 and HOA-2) were designed. The local search in HOA-1 is a type of pattern search with random factors, and that of HOA-2 is based on the sampling in the iteratively updated Gaussian distributions. Considering the limitations of regular benchmark functions, twenty-three variant benchmark functions were designed and solved to accurately reflect the performance of the proposed algorithms and compared algorithms. The experiment results show that HOA-1 outperforms other algorithms in the benchmark function test. In order to further verify the feasibility and superiority of the proposed algorithms, two spacecraft trajectory optimization problems, multiple-impulse orbit transfer and spacecraft pursuit-evasion game, were introduced, due to their complexity and challenges. The HOA-2 algorithm excels outstandingly in solving trajectory optimization problems. In HOA-1, the number of S-particles should be identical to the dimension of the problem, which partially limits its performance and adaptivity.

The parameter T greatly influences the performance of proposed algorithms, especially on the convergence speed. Generally, a larger T is suitable to solve the unimodal function, while a smaller T is preferred in solving the multimodal function. Also, the allocation of function evaluation numbers between the B-particles and S-particles is important to determine the search efficiency. For future research, searching strategy design and techniques of parameter setting are both promising research directions for promoting the development of hierarchical optimization.

Author Contributions: Conceptualization, H.H. and P.S.; methodology, H.H. and P.S.; software, H.H.; investigation, H.H.; writing—original draft preparation, H.H.; writing—review and editing, P.S

and Y.Z.; funding acquisition, P.S. All authors have read and agreed to the published version of the manuscript.

Funding: This work is supported by the National Natural Science Foundation of China (No. 11572019).

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

Appendix A. Benchmark Functions

Table A1. Unimodal benchmark functions.

Function	Dim	Range	$f_{\min}$
$F1 = \sum_{i=1}^{\mathrm{Dim}} x_i^2$	30	$[-100, 100]$	0
$F2 = \sum_{i=1}^{\mathrm{Dim}} \lvert x_i \rvert + \prod_{i=1}^{\mathrm{Dim}} \lvert x_i \rvert$	30	$[-10, 10]$	0
$F3 = \sum_{i=1}^{\mathrm{Dim}} \left(\sum_{j=1}^{i} x_j \right)^2$	30	$[-100, 100]$	0
$F4 = \{ \lvert x_i \rvert, 1 \le i \le \mathrm{Dim} \}$	30	$[-100, 100]$	0
$F5 = \sum_{i=1}^{\mathrm{Dim}} \left(100(x_{i+1} - x_i^2)^2 + (x_i + 1)^2 \right)$	30	$[-30, 30]$	0
$F6 = \sum_{i=1}^{\mathrm{Dim}} \left(\lvert x_i + 0.5 \rvert \right)^2$	30	$[-100, 100]$	0
$F7 = \sum_{i=1}^{\mathrm{Dim}} i x_i^4 + random[0, 1)$	30	$[-1.28, 1.28]$	0

Table A2. Multimodal benchmark functions.

Function	Dim	Range	$f_{\min}$
$F8 = \sum_{i=1}^{\mathrm{Dim}} -x_i \sin(\sqrt{\lvert x_i \rvert})$	30	$[-500, 500]$	$-419 \times \mathrm{Dim}$
$F9 = \sum_{i=1}^{\mathrm{Dim}} \left[x_i^2 - 10\cos(2\pi x_i) + 10 \right]$	30	$[-5.12, 5.12]$	0
$F10 = -20\exp\left(-0.2\sqrt{\sum_{i=1}^{\mathrm{Dim}} x_i^2 / \mathrm{Dim}} \right) - \exp\left(\sum_{i=1}^{\mathrm{Dim}} \cos(2\pi x_i) / \mathrm{Dim} \right) + 20 + e$	30	$[-32, 32]$	0
$F11 = \sum_{i=1}^{\mathrm{Dim}} x_i^2 / 4000 - \prod_{i=1}^{\mathrm{Dim}} \cos\left(x_i / \sqrt{i} \right) + 1$	30	$[-600, 600]$	0
$F12 = \frac{\pi}{\mathrm{Dim}} \left\{ \sin(\pi y_1) + \sum_{i=1}^{\mathrm{Dim}-1} (y_i - 1)^2 [1 + 10\sin^2(\pi y_{i+1})] + (y_{\mathrm{Dim}} - 1)^2 \right\} + \sum_{1}^{\mathrm{Dim}} u(x_i, 10, 100, 4)$, $y_i = 1 + \frac{x_i + 1}{4}$, $u(x_i, a, k, m) = \begin{cases} k(x_i - a)^m & x_i > a \\ 0 & -a < x_i < a \\ k(-x_i - a)^m & x_i < -a \end{cases}$	30	$[-50, 50]$	0
$F13 = 0.1 \left\{ \begin{array}{l} \sin^2(3\pi x_1) + \sum_{i=1}^{\mathrm{Dim}-1} (x_i - 1)^2 [1 + \sin^2(3\pi x_{i+1})] \\ + (x_{\mathrm{Dim}} - 1)^2 [1 + \sin^2(2\pi x_{\mathrm{Dim}})] \end{array} \right\} + \sum_{1}^{\mathrm{Dim}} u(x_i, 5, 100, 4)$	30	$[-50, 50]$	0

Table A3. Fixed-dimension benchmark functions.

Function	Dim	Range	$f_{\min}$
$F14 = \left(\frac{1}{500} + \sum_{j=1}^{25} \frac{1}{j + \sum_{i=1}^{2} (x_i - a_{ij})^6} \right)^{-1}$	2	$[-65, 65]$	0.998
$F15 = \sum_{i=1}^{11} \left[a_i - \frac{x_1 (b_i^2 + b_i x_2)}{b_i^2 + b_i x_3 + x_4} \right]^2$	4	$[-5, 5]$	0.0003
$F16 = 4x_1^2 - 2.1x_1^4 + \frac{1}{3}x_1^6 + x_1 x_2 - 4x_2^2 + 4x_2^4$	2	$[-5, 5]$	-1.0316
$F17 = \left(x_2 - \frac{5.1}{4\pi^2} x_1^2 + \frac{5}{\pi} x_1 - 6 \right)^2 + 10\left(1 - \frac{1}{8\pi} \right) \cos x_1 + 10$	2	$[-5, 5]$	0.398
$F18 = \left[1 + (x_1 + x_2 + 1)^2 (19 - 14x_1 + 3x_1^2 - 14x_2 + 6x_1 x_2 + 3x_2^2) \right] \times \left[30 + (2x_1 - 3x_2)^2 (18 - 32x_1 + 12x_1^2 + 48x_2 - 36x_1 x_2 + 27x_2^2) \right]$	2	$[-2, 2]$	3

Table A3. *Cont.*

Function	Dim	Range	$f_{\min}$
F19 $= -\sum_{i=1}^{4} c_i \exp\left(-\sum_{j=1}^{3} a_{ij}\left(x_j - p_{ij}\right)^2\right)$	3	[0, 1]	−3.86
F20 $= -\sum_{i=1}^{4} c_i \exp\left(-\sum_{j=1}^{6} a_{ij}\left(x_j - p_{ij}\right)^2\right)$	6	[0, 1]	−3.32
F21 $= \sum_{i=1}^{5} \left[(X - a_i)^T (X - a_i) + c_i\right]^{-1}$	4	[0, 10]	−10.1532
F22 $= \sum_{i=1}^{7} \left[(X - a_i)^T (X - a_i) + c_i\right]^{-1}$	4	[0, 10]	−10.4028
F23 $= \sum_{i=1}^{10} \left[(X - a_i)^T (X - a_i) + c_i\right]^{-1}$	4	[0, 10]	−10.5363

References

1. Zhang, P.; Xu, Z.; Wang, Q.; Fan, S.; Cheng, W.; Wang, H.; Wu, Y. A novel variable selection method based on combined moving window and intelligent optimization algorithm for variable selection in chemical modeling. *Spectrochim. Acta Part A Mol. Biomol. Spectrosc.* **2021**, *246*, 118986. [CrossRef]
2. Lin, C.; Wang, J.; Lee, C. Pattern recognition using neural-fuzzy networks based on improved particle swam optimization. *Expert Syst. Appl.* **2009**, *36*, 5402–5410. [CrossRef]
3. Majeed, A.; Hwang, S. A Multi-Objective Coverage Path Planning Algorithm for UAVs to Cover Spatially Distributed Regions in Urban Environments. *Aerospace* **2021**, *8*, 343. [CrossRef]
4. Gao, X.Z.; Nalluri, M.S.R.; Kannan, K.; Sinharoy, D. Multi-objective optimization of feature selection using hybrid cat swarm optimization. *Sci. China Technol. Sci.* **2021**, *64*, 508–520. [CrossRef]
5. Singh, T. A novel data clustering approach based on whale optimization algorithm. *Expert Syst.* **2021**, *38*, e12657. [CrossRef]
6. Pontani, M.; Ghosh, P.; Conway, B. Particle swarm optimization of multiple-burn rendezvous trajectories. *J. Guid. Control Dyn.* **2012**, *35*, 1192–1207. [CrossRef]
7. Wagner, S.; Wie, B. Hybrid algorithm for multiple gravity-assist and impulsive delta-V maneuvers. *J. Guid. Control Dyn.* **2015**, *38*, 2096–2107. [CrossRef]
8. Wang, X.; Zhang, H.; Bai, S.; Yue, Y. Design of agile satellite constellation based on hybrid-resampling particle swarm optimization method. *Acta Astronaut.* **2021**, *178*, 595–605. [CrossRef]
9. Wu, C.; Xu, R.; Zhu, S.; Cui, P. Time-optimal spacecraft attitude maneuver path planning under boundary and pointing constraints. *Acta Astronaut.* **2017**, *137*, 128–137. [CrossRef]
10. Mirjalili, S.; Mirjalili, S.M.; Lewis, A. Grey wolf optimizer. *Adv. Eng. Softw.* **2014**, *69*, 46–61. [CrossRef]
11. Holland, J.H. Genetic algorithms. *Sci. Am.* **1992**, *267*, 66–72. [CrossRef]
12. Yao, X.; Liu, Y.; Lin, G. Evolutionary programming made faster. *IEEE Trans. Evol. Comput.* **1999**, *3*, 82–102. [CrossRef]
13. Storn, R.; Price, K. Differential evolution—A simple and efficient heuristic for global optimization over continuous spaces. *J. Glob. Optim.* **1997**, *11*, 341–359. [CrossRef]
14. Juang, C. A hybrid of genetic algorithm and particle swarm optimization for recurrent network design. *IEEE Trans. Syst. Man. Cybern.* **2004**, *34*, 997–1006. [CrossRef] [PubMed]
15. Zhu, X.; Zhang, H.; Gao, Y. Correlations between the scaling factor and fitness values in differential evolution. *IEEE Access.* **2020**, *8*, 32100–32120. [CrossRef]
16. Smith, J. Coevolving memetic algorithms: A review and progress report. *IEEE Trans. Syst. Man Cybern. Part B-Cybern.* **2007**, *37*, 6–17. [CrossRef] [PubMed]
17. Smith, J.; Fogarty, T. Operator and parameter adaptation in genetic algorithms. *Soft Comput.* **1997**, *1*, 81–87. [CrossRef]
18. Kennedy, J.; Eberhart, R. Particle swarm optimization. In Proceedings of the 1995 IEEE International Conference on Neural Networks, Perth, WA, USA, 27 November–1 December 1995; pp. 1942–1948. [CrossRef]
19. Shi, Y.; Eberhart, R. A modified particle swarm optimizer. In Proceedings of the 1998 IEEE Word Congress on Computational Intelligence, Anchorage, AK, USA, 4–9 May 1998; pp. 69–73. [CrossRef]
20. Kennedy, J.; Eberhart, R. A discrete binary version of the particle swarm algorithm. In Proceedings of the 1997 IEEE International Conference on Systems, Man, and Cybernetics-Computational Cybernetics and Simulation, Orlando, FL, USA, 12–15 October 1997; pp. 4104–4108. [CrossRef]
21. Rana, S.; Jasola, S.; Kumar, R. A review on particle swarm optimization algorithms and their applications to data clustering. *Artif. Intell. Rev.* **2011**, *35*, 211–222. [CrossRef]
22. Yang, X. Firefly algorithm, stochastic test functions and design optimization. *Int. J. Bio-Inspired Comput.* **2010**, *2*, 78–84. [CrossRef]
23. Heidari, A.A.; Mirjalili, S.; Faris, H.; Aljarah, I.; Mafarja, M.; Chen, H. Harris hawks optimization: Algorithm and applications. *Future Gener. Comput. Syst.* **2019**, *97*, 849–872. [CrossRef]
24. Gao, K.; Cao, Z.; Zhang, L.; Han, Y.; Pan, Q. A review on swarm intelligence and evolutionary algorithms for solving flexible job shop scheduling problems. *IEEE-CAA J. Autom. Sin.* **2019**, *6*, 904–916. [CrossRef]
25. Kirkpatrick, S.; Gelatt, C.D.; Vecchi, M.P. Optimization by simulated annealing. *Science* **1983**, *220*, 671–680. [CrossRef] [PubMed]
26. Rashedi, E.; Nezamabadi-Pour, H.; Saryazdi, S. GSA: A gravitational search algorithm. *Inf. Sci.* **2009**, *179*, 2232–2248. [CrossRef]

27. Mirjalili, S. SCA: A Sine Cosine Algorithm for solving optimization problems. *Knowl.-Based Syst.* **2016**, *96*, 120–133. [CrossRef]
28. Abualigah, L.; Diabat, A.; Mirjalili, S.; Elaziz, M.A.; Gandomi, A.H. The arithmetic optimization algorithm. *Comput. Methods Appl. Mech. Engrgy* **2021**, *376*, 113609. [CrossRef]
29. Wolpert, D.; Macready, W.G. No free lunch theorems for optimization. *IEEE Trans. Evolut. Comput.* **1997**, *1*, 67–82. [CrossRef]
30. Passino, K. Biomimicry of bacterial foraging for distributed optimization and control. *IEEE Control Syst. Mag.* **2002**, *22*, 52–67. [CrossRef]
31. Jin, M.; Lu, H. A multi-subgroup hierarchical hybrid of genetic algorithm and particle swarm optimization. *Control Theory Appl.* **2013**, *30*, 1231–1238. [CrossRef]
32. Liu, L.; Guo, Y. Multi-objective optimization for attitude maneuver of liquid-filled flexible spacecraft based on improved hierarchical optimization algorithm. *Appl. Soft Comput.* **2020**, *96*, 106598. [CrossRef]
33. Vasile, M.; Locatelli, M. A hybrid multiagent approach for global trajectory optimization. *J. Glob. Optim.* **2009**, *44*, 461–479. [CrossRef]
34. Zuiani, F.; Vasile, M. Multi Agent Collaborative Search based on Tchebycheff decomposition. *Comput. Optim. Appl.* **2013**, *56*, 189–208. [CrossRef]
35. Arab, A.; Alfi, A. An adaptive gradient descent-based local search in memetic algorithm applied to optimal controller design. *Inf. Sci.* **2015**, *299*, 117–142. [CrossRef]
36. Bao, Y.; Hu, Z.; Xiong, T. A PSO and pattern search based memetic algorithm for SVMs parameters optimization. *Neurocomputing* **2013**, *117*, 98–106. [CrossRef]
37. Lin, S. A parallel processing multi-coordinate descent method with line search for a class of large-scale optimization-algorithm and convergence. In Proceedings of the 30th IEEE Conference on Decision and Control, Brighton, UK, 11–13 December 1991. [CrossRef]
38. Kennedy, J. Bare bones particle swarms. In Proceedings of the 2003 IEEE Swarm Intelligence Symposium, Indianapolis, IN, USA, 26 April 2003; pp. 80–87. [CrossRef]
39. Zhang, E.; Wu, Y.; Chen, Q. A practical approach for solving multi-objective reliability redundancy allocation problems using extended bare-bones particle swarm optimization. *Reliab. Eng. Syst. Saf.* **2014**, *127*, 65–76. [CrossRef]
40. Zhang, Y.; Gong, D.; Ding, Z. A bare-bones multi-objective particle swarm optimization algorithm for environmental/economic dispatch. *Inf. Sci.* **2012**, *192*, 213–227. [CrossRef]
41. Hansen, N.; Muller, S.; Koumoutsakos, P. Reducing the time complexity of the derandomized evolution strategy with covariance matrix adaptation (CMA-ES). *Evol. Comput.* **2003**, *11*, 1–18. [CrossRef]
42. Arora, S.; Singh, S. Butterfly optimization algorithm: A novel approach for global optimization. *Soft Comput.* **2019**, *23*, 715–734. [CrossRef]
43. Derrac, J.; García, S.; Molina, D.; Herrera, F. A practical tutorial on the use of nonparametric statistical tests as a methodology for comparing evolutionary and swarm intelligence algorithms. *Swarm Evol. Comput.* **2011**, *1*, 3–18. [CrossRef]
44. Abdelkhalik, O.; Mortari, D. N-impulse orbit transfer using genetic algorithms. *J. Spacecr. Rockets* **2007**, *44*, 456–460. [CrossRef]
45. Wang, X.; Shi, P.; Zhao, Y.; Sun, Y. A pre-trained fuzzy reinforcement learning method for the pursuing satellite in a one-to-one game in space. *Sensors* **2020**, *20*, 2253. [CrossRef]
46. Hughes, S.; Mailhe, L.; Guzman, J. A comparison of trajectory optimization methods for the impulsive minimum fuel rendezvous problem. *Adv. Astronaut. Sci.* **2003**, *113*, 85–104.
47. Curtis, H. *Orbital Mechanics for Engineering Students*; Elsevier Butterworth-Heinemann: Oxford, UK, 2010; pp. 257–264. [CrossRef]
48. Yang, Z.; Luo, Y.; Zhang, J. Nonlinear semi-analytical uncertainty propagation of trajectory under impulsive maneuvers. *Astrodynamics* **2019**, *3*, 61–77. [CrossRef]
49. Sun, S.; Zhang, Q.; Loxton, R.; Li, B. Numerical solution of a pursuit-evasion differential game involving two spacecraft in low Earth orbit. *J. Ind. Manag. Optim.* **2015**, *11*, 1127–1147. [CrossRef]
50. Clohessy, W.; Wiltshire, R. Terminal guidance system for satellite rendezvous. *J. Aerosp. Sci.* **1960**, *11*, 653–658. [CrossRef]