Computers, Materials & Continua

Tech Science Press

https://doi.org/10.32604/cmc.2026.083353

ARTICLE

A Two-Stage, Nested Co-Optimization Framework with Adaptive Evolutionary Operators for Component-Level Constellation Morphology and Mission Planning

Chao Zhang and Yunfeng Dong*

School of Astronautics, Beihang University, Beijing, China

*Corresponding Author: Yunfeng Dong. Email: sinosat@buaa.edu.cn

Received: 02 April 2026; Accepted: 17 June 2026; Published: 13 August 2026

ABSTRACT: The missile warning constellation is fundamental to national territorial security and has significant strategic and military value. This study proposes a two-stage, nested co-optimization framework with adaptive evolutionary operators, termed TNC-A, to address challenges in genetic representation, evaluation distortion, the curse of dimensionality, and search inefficiency within the co-optimization of component-level constellation morphology and mission planning. A hybrid encoding scheme combining tree-structured and real-valued vector representations was adopted to encode all optimization variables, including constellation configuration, component-level unified platform information, and mission planning parameters. Second, a multi-stage optimization strategy integrated with a double-nested structure was implemented. This approach mitigates the dimensionality curse and addresses the evaluation distortion inherent in traditional engineering methods. Third, adaptive evolutionary operators based on online contribution learning were designed to dynamically adjust the search focus using historical experience, thereby significantly enhancing search efficiency. The simulation results demonstrate the effectiveness and practical applicability of TNC-A, demonstrating its superiority over comparative algorithms under a constrained function evaluation budget. TNC-A reduced the average and standard deviation of the optimization results by 13.7% and 65.3%, respectively, compared with the baseline of traditional engineering methods.

KEYWORDS: Co-optimization; double-nested optimization structure; multi-stage optimization strategy; adaptive evolutionary operators; component-level; missile warning constellation

1 Introduction

In modern missile defense systems, missile warning constellations have become the core of strategic early-warning systems due to their global coverage, continuous monitoring, and rapid response [1]. These constellations are primarily responsible for detecting, identifying, and continuously tracking incoming missiles; their performance directly determines the available interception window and the probability of successful defense [2]. Consequently, they have substantial strategic and military value.

Optimization of missile warning constellations is critical because it directly determines their performance [3]. This optimization process involves the intricate coupling of hardware design (e.g., platform and payload component selection) and software design (e.g., mission planning). The full potential of a missile warning constellation can only be realized when hardware and software designs are properly harmonized. Accordingly, this study addresses the co-optimization problem of missile warning constellation morphology and mission planning to ensure global optimality and operational feasibility.

Constellation morphology optimization and mission planning are generally treated as two independent subproblems [4]. At the constellation morphology optimization level, most studies have primarily focused on system-level constellation optimization (including constellation configuration parameters such as the orbital elements of each satellite) using intelligent algorithms, often abstracting each satellite as a point mass [5,6]. Limited research has addressed constellation morphology optimization at the component-level granularity, where morphology encompasses not only configuration parameters but also component selection and subsystem parameters for each satellite. Component-level constellation morphology optimization provides more detailed and higher-value results than system-level granularity. At the constellation mission planning level, distributed mission planning approaches [7–9] have gradually supplanted traditional centralized approaches [10,11] as the mainstream, owing to their superior flexibility, scalability, and reduced computational complexity. The contract network protocol (CNP) [12], which relies on a set of predefined rules to guide mission allocation and collaboration among satellites, is a core mechanism of distributed planning. The parameters governing these rules significantly influence the performance of mission planning; however, they are typically treated as fixed values rather than optimization variables. The optimal parameters are highly dependent on the specific constellation morphology, requiring tailored configurations for each morphology. Consequently, the performance of missile warning constellations is determined by the hardware foundation provided by their morphology, while mission planning governs the effective use of this hardware. This highlights the deep coupling between constellation morphology and mission planning and the need for their co-optimization. However, the extreme complexity of the co-optimization problem introduces four major challenges. The first difficulty is in genetic representation. The optimization variables involved are extremely complex, mainly encompassing component-level constellation morphology with discrete hierarchical structures (e.g., constellation → satellite → subsystem → component) and continuous mission planning parameters, all of which substantially increase the problem encoding complexity. The second is the curse of dimensionality. The extreme complexity of the optimization variables results in an exponentially expanding solution space, making direct search typically infeasible. The third factor is evaluation distortion. Typically, traditional engineering methods adopt a sequential optimization structure. This implies optimizing constellation morphology before mission planning parameters. Consequently, morphology schemes are evaluated under non-optimal mission planning parameters, failing to reflect their true potential and potentially misleading the search direction. The fourth factor is search inefficiency. The extreme complexity of the co-optimization problem significantly enlarges the solution space and increases variable coupling, resulting in substantially reduced search efficiency and slow convergence.

To address the difficulty in genetic representation, hybrid encoding schemes [13] have emerged as an effective solution. Specifically, a tree-structured representation [14] can encode optimization variables with distinct hierarchical structures, whereas a real-valued vector representation adopted in the optimization framework proposed in [15] can encode the remaining continuous optimization variables. Therefore, the combination of tree-structured and real-valued vector representations provides an effective solution to this challenge.

To mitigate the curse of dimensionality, multi-stage optimization strategies [16] have been recognized as an effective solution, as they can significantly reduce the complexity of the co-optimization problem. However, due to the extreme complexity of the co-optimization problem, the reasonable design of stage division and stage switching strategies remains a significant challenge.

To overcome evaluation distortion, inspiration can be drawn from robot morphology and control co-optimization [17], where nested optimization structures have been shown to effectively decouple robot morphology and control variables. Such structures can decouple constellation morphology optimization from mission planning optimization, thereby overcoming the evaluation distortion inherent in sequential

optimization structures. However, organically integrating the nested optimization structure with the multi-stage optimization strategy into a unified framework remains a significant challenge.

To address search inefficiency, current research has primarily focused on the design and enhancement of evolutionary operators [18–20]. However, the extreme complexity of the co-optimization problem, coupled with the aforementioned hybrid encoding schemes (including tree-structured and real-valued vector representations), presents significant difficulties in designing and enhancing evolutionary operators. Traditional evolutionary operators are generally unable to effectively perceive the complex topological characteristics arising from the combination of tree-structured and real-valued vector representations, resulting in blind exploration within a high-dimensional solution space and extremely slow convergence.

To address these challenges, this study proposes TNC-A, with the following main contributions.

(1) A multi-stage optimization strategy combined with a double-nested optimization structure is designed. In TNC-A, the optimization process is divided into two stages based on practical engineering experience: the first stage explores macroscopic constellation morphology, while the second focuses on the detailed, component-level design of each satellite within the constellation. A double-nested optimization structure is applied within each stage, ensuring that each constellation morphology is evaluated under its optimal mission planning parameters. In addition, a stage-switching strategy enables individuals to flexibly transition between stages according to actual conditions.

(2) Adaptive evolutionary operators are designed. TNC-A incorporates adaptive evolutionary operators based on online contribution learning to maximize search efficiency. Leveraging the hybrid encoding scheme, which combines tree-structured and real-valued vector representations, custom evolutionary operators are designed to accommodate the unique search space. The adaptive mechanism dynamically adjusts the search focus based on the historical performance data acquired through online contribution learning.

The remainder of this study is organized as follows. The problem description of the component-level constellation morphology and mission planning co-optimization is given in Section 2. TNC-A is described in Section 3. The simulation results and discussion are provided in Section 4. The conclusions are presented in Section 5.

2 Problem Description

The goal of this study is to seek the optimal pairing of constellation morphology and mission planning, enabling practical, component-level recommendations for constellation morphology design. The optimization problem is described as follows:

$$\begin{aligned} &\text{find} && \boldsymbol{S}, \\ &\min && F(\boldsymbol{X}), \\ &&& \boldsymbol{S} \rightarrow M(\boldsymbol{X}), \\ &\text{s.t.} && d\boldsymbol{X}/dt = M(\boldsymbol{X}), \boldsymbol{X}(t_0) = \boldsymbol{X}_0, \\ &&& \boldsymbol{Con}(\boldsymbol{X}) \geq 0, \end{aligned} \tag{1}$$

where $\boldsymbol{S}$ is the optimization variable set, $F(\boldsymbol{X})$ is the objective function. $\boldsymbol{S} \rightarrow M(\boldsymbol{X})$ represents the automatic establishment of the missile warning constellation model (named the MWC model), which is described in Section 2.3.3. $\boldsymbol{X}$ and $d\boldsymbol{X}/dt = M(\boldsymbol{X}), \boldsymbol{X}(t_0) = \boldsymbol{X}_0$ are the state variable set and state equation of the MWC model, respectively. $\boldsymbol{Con}(\boldsymbol{X}) \geq 0$ represents all constraints.

2.1 Optimization of the Variable Set

Each optimization variable set $\boldsymbol{S}$ represents a specific scheme that encompasses both constellation morphology and mission planning information.

Constellation morphology information comprises constellation configuration and component-level platform details.

The constellation configuration information describes the satellite positions and payload selection for each satellite. In practical engineering applications, satellites within a constellation predominantly adopt a unified platform design to maximize resource utilization, minimize costs, and ensure sustainable long-term operation and maintenance. Consequently, all satellites within a constellation share a unified platform, with only the payloads (specifically, the cameras in this study) varying according to specific requirements. Therefore, each satellite is assumed to have a single platform option but multiple selectable camera models. Furthermore, satellite positions are designed based on the Walker constellation because of its robust global coverage capability [21]. The optimization variables constituting the constellation configuration information are defined as follows

$$cs_1, cs_2, cs_3, cs_4, cs_5, cp_i \in \left[1, 2, 3, \ldots, N_{\text{pay}}\right], 1 \leq i \leq N_{\text{sat}}, \tag{2}$$

where cs_1 is the number of orbital planes, cs_2 is the number of satellites per orbital plane, cs_3 is the orbital inclination, cs_4 is the orbital altitude and cs_5 is the phasing factor. cp_i represents the camera model index selected for the i-th satellite, N_{pay} is the total number of available camera models, and N_{sat} represent the total number of satellites in the constellation.

Component-level information describes the composition, structure, and parameters of the unified platform based on [14,22], as shown in Fig. 1. The platform comprises six subsystems: the structure subsystem (SS), attitude and orbit control subsystem (AOCS), power subsystem (PoS), thermal subsystem (TS), telemetry, tracking, and command subsystem (TTCS), and propulsion subsystem (PrS).

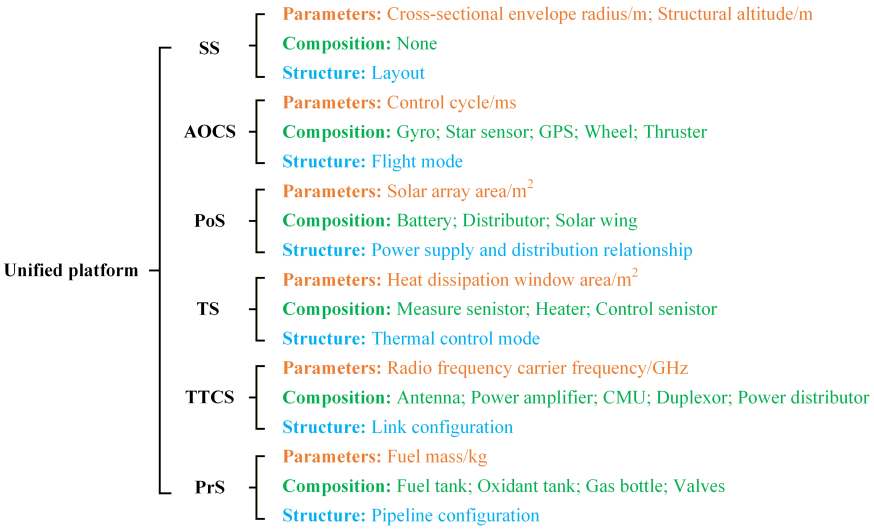

Figure 1: Typical unified platform composition, structure, and parameters.

The unified platform composition, structure and parameter information can be represented as a tree structure $\boldsymbol{st}$ with reference to the coding method of the GP [22,23], as seen in Fig. 2.

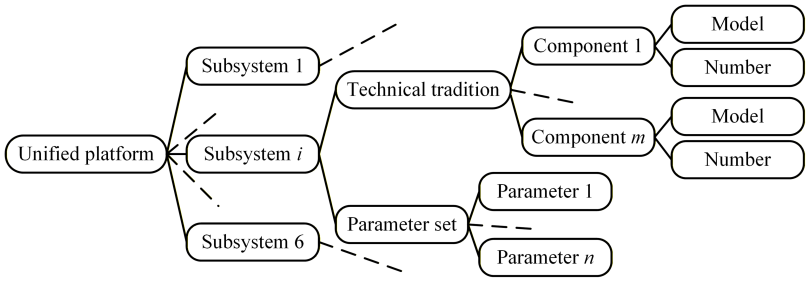

Figure 2: Tree structure.

Mission planning information quantifies the relative emphasis on assigning identification and tracking missions. Specifically, it comprises the identification mission coefficient cm_1 and the tracking mission coefficient cm_2, which are detailed in Section 2.3.2. In this study, both cm_1 and cm_2 assume values from the discrete set $\{0.1, 0.2, \ldots, 1.0\}$.

Collectively, $\boldsymbol{S}$ encompasses $N_{\text{sat}} + 8$ optimization variables: cs_1, cs_2, cs_3, cs_4, cs_5, cp_i $(1 \le i \le N_{\text{sat}})$, $\boldsymbol{st}$, cm_1 and cm_2.

2.2 Objective Functions and Constraints

The evaluation of a scheme must comprehensively consider three primary factors: performance, reliability, and cost. In practical engineering applications, reliability and cost are generally subject to strict regulatory requirements and budgetary limitations; therefore, they function as hard constraints that must be satisfied. Meanwhile, performance serves as the objective function $F(\boldsymbol{X})$.

$F(\boldsymbol{X})$ is calculated by simulating the MWC model, as follows:

$$F(\boldsymbol{X}) = 1 - \left(\omega_1 \cdot \left(t_{\text{total}} - t_{\text{LSI}}\right)/t_{\text{total}} + \omega_2 \cdot N_{\text{SI}}/N_{\text{missile}}\right), \tag{3}$$

where ω_1 and ω_2 are two weight coefficients. t_{total} is the total simulation time of the MWC model, t_{LSI} is the time when the defensive force successfully intercepts the incoming missiles launched by the offensive force for the last time, N_{SI} is the total number of incoming missiles successfully intercepted by the defensive force, N_{missile} is the total number of incoming missiles launched by the offensive force. In this study, the defensive force aims to maximize missile interceptions (corresponding to $N_{\text{SI}}/N_{\text{missile}}$) while minimizing the engagement duration (corresponding to $\left(t_{\text{total}} - t_{\text{LSI}}\right)/t_{\text{total}}$), such that the calculation of $F(\boldsymbol{X})$ is directly dependent on t_{LSI} and N_{SI}.

The constraints are defined as follows:

$$\boldsymbol{Con}(\boldsymbol{X}) \ge 0 \Leftrightarrow \begin{cases} N_{\text{sat}} \le N_{\text{s}} \\ C(\boldsymbol{X}) \le C_{\text{Req}} \\ R(\boldsymbol{X}) \le R_{\text{Req}} \end{cases}, \tag{4}$$

where N_{s} denotes the maximum total number of satellites within the constellation. The constraint $N_{\text{sat}} \le N_{\text{s}}$ is imposed to ensure the tractability of the problem encoding process, as detailed in Section 3.1. C_{Req} and R_{Req} represent the maximum allowable cost and reliability, respectively. In practical engineering applications, C_{Req} and R_{Req} are determined through a comprehensive assessment of design specifications, operational requirements, budgetary constraints, and risk tolerance.

$C(\boldsymbol{X})$ is the cost of the constellation, which is calculated as follows:

$$C(\boldsymbol{X}) = \left(\sum_{i=1}^{N_{\text{sat}}} \left(C_i^{\text{sat}}\right) - C_{\min}\right) / \left(C_{\max} - C_{\min}\right), \tag{5}$$

where $C_{\max}$ and $C_{\min}$ are maximum and minimum limits of cost, respectively. C_i^{sat} is the cost of satellite i, which is calculated based on the unmanned space vehicle cost model [24].

$R(\boldsymbol{X})$ is the reliability of the constellation, which is calculated as follows:

$$R(\boldsymbol{X}) = \left(R_{\max} - \left(\sum_{i=1}^{N_{\text{sat}}} \left(R_i^{\text{sat}}\right) / N_{\text{sat}}\right)\right) / \left(R_{\max} - R_{\min}\right), \tag{6}$$

where $R_{\max}$ and $R_{\min}$ are maximum and minimum limits of reliability, respectively. R_i^{sat} is the reliability of satellite i, which is calculated based on the literature [25].

2.3 Automatic Construction of the MWC Model

The MWC model (Table 1) simulates offensive and defensive missile warning operations, as detailed below. The offensive force aims to destroy all defensive military bases through missile strikes, whereas the defensive force seeks to protect its bases by intercepting incoming missiles to the greatest possible extent. Specifically, the offensive command center issues attack orders, prompting missile units to launch strikes against defensive bases. The defensive force relies on its missile warning constellation to detect, identify, and continuously track incoming missiles. The resulting fire control data are transmitted to defensive bases, which then launch interception missiles based on this information.

Table 1: MWC model.

The MWC Model		Description
Flight environment sub-model		Constructed with reference to the multi-scale multi-physics field-coupled complex model [14].
Inter-satellite communication sub-model		Described as follows.
Satellite sub-model	Unified platform sub-model	Constructed with reference to the component prototype model [14].
	Camera sub-model	
Missile sub-model		Constructed with reference to the hypersonic glide vehicle dynamic model [26].
Interception planning sub-model		Described in Section 2.3.1.
Mission planning sub-model		Described in Section 2.3.2.

In the inter-satellite communication sub-model, multiple factors, including satellite position, antenna attitude, antenna gain, axial ratio, and polarization angle, determine the communication status between any two satellites.

2.3.1 Interception Planning Sub-Model

In the interception planning sub-model, the greedy algorithm [27] is adopted as the core scheduling method due to its structural simplicity and high computational efficiency in practical engineering applications. Once an incoming missile is successfully tracked, the interceptor with the shortest interception time is prioritized for assignment.

2.3.2 Mission Planning Sub-Model

The missile warning constellation is responsible for all incoming missile discovery, identification, and tracking. Three mission types are defined: discovery, identification, and tracking. Discovery missions are restricted to satellites equipped with wide-field cameras that scan potential areas to locate missiles with uncertain positions. Identification and tracking missions are assigned to satellites equipped with high-resolution cameras. During identification missions, satellites observe detected missiles in detail to determine their specific types, models, and technical parameters. In tracking missions, satellites continuously track identified missiles and provide fire control data to military bases.

The Contract Net Protocol (CNP) is employed as the core scheduling method in the mission planning sub-model due to its structural simplicity, strong adaptability, and high robustness [12]. Compared with centralized approaches, the autonomy and low-latency characteristics of CNP are essential for rapid response without ground intervention, which is critical for minimizing overall warning and response times. The mission planning process comprises three steps.

Step 1: Calculate the visible time windows between satellites and missiles based on their positions during the scheduling period. Each visible time window represents a potential mission.

Step 2: Calculate the profits and costs of all missions. The profit calculation for a mission depends on its type, as described below:

$$Profit_i = \begin{cases} Bat_k \cdot cm_1 \cdot (t - t_k) / T_{\text{lost}}, & \text{type} \in \text{identification mission} \\ Bat_k \cdot cm_2 \cdot (t - t_k) / T_{\text{lost}}, & \text{type} \in \text{tracking mission} \end{cases} \tag{7}$$

The cost of a single mission is calculated as follows:

$$Cost_i = tw_i^{\text{end}} - t, \tag{8}$$

where $Profit_i$ and $Cost_i$ are profit and cost of the i-th mission, respectively. The i-th mission is regarded as the i-th visible time window between the j-th satellite and the k-th missile. Bat_k is the base profit of the k-th missile. T_{lost} is the loss threshold: if a missile remains unobserved by any satellite for a continuous period exceeding T_{lost}, it will be defined as a lost state. t is the current simulation time, and t_k is the last simulation time at which the k-th missile was successfully observed. tw_i^{end} is the end time of the i-th visible time window. cm_1 and cm_2 are weighting coefficients that represent strategic preferences between identification and tracking missions. These coefficients directly affect the profit calculation for each mission, thereby influencing the assignment priorities of different missions during the mission planning process (Step 3).

Step 3: Perform mission planning. The remaining mission with the highest profit is prioritized and assigned to the idle satellite with the lowest cost that satisfies all constraints.

Step 4: Execute missions. The planning results, including attitude maneuver and imaging instructions, are transmitted to the relevant satellites, which then execute the assigned missions.

2.3.3 Automatic Construction Technology

Automatic construction technology [14] is employed to construct the MWC models. The digital satellite software developed by our laboratory is the core of this technology, which automatically generates the source code for the MWC models.

3 TNC-A

The TNC-A process is shown in Fig. 3.

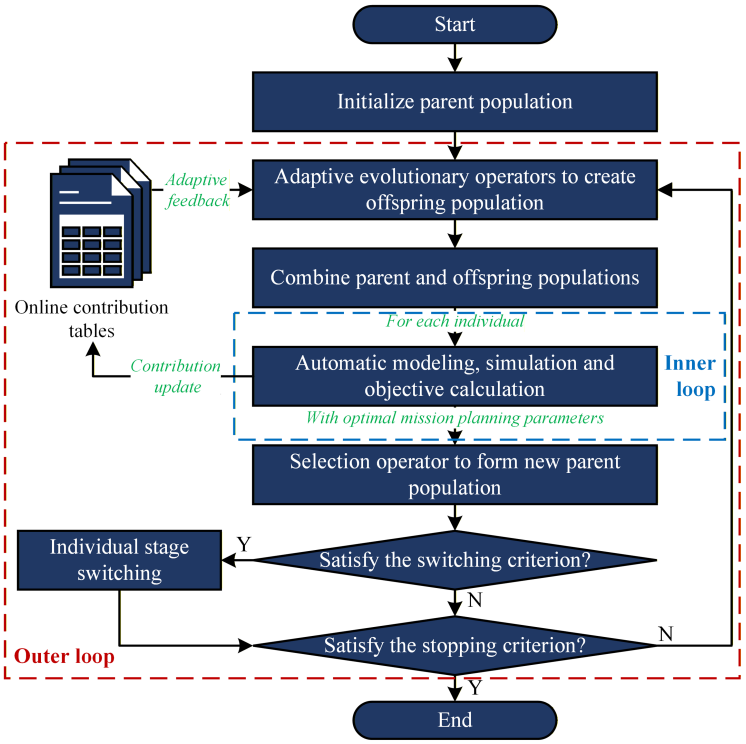

Figure 3: TNC-A process.

As shown in Fig. 3, TNC-A terminates based on one of two stopping criteria: reaching the maximum number of iterations N_{MaxI} or exceeding the maximum allowable number of function evaluations N_{MaxFE}. The double-nested optimization structure comprises the outer and inner loops. The outer loop explores the constellation morphology, whereas the inner loop identifies the optimal mission planning parameters corresponding to each constellation morphology.

3.1 Problem Encoding

A hybrid encoding scheme that combines tree-structured and real-valued vector representations is employed given the specific nature of the problem, as illustrated in Fig. 4.

The optimization variables described in Section 2.1 are encoded into genetic representations for each individual in Fig. 4, with component-level platform information structured as a tree structure $\boldsymbol{st}$. Each individual undergoes a two-stage optimization process that incorporates outer and inner loops. The specific optimization focus for each stage is summarized in Table 2. In this table, 'Changeable' indicates that the corresponding genetic codes participate in the optimization, whereas 'Unchangeable' signifies that they are treated as constants.

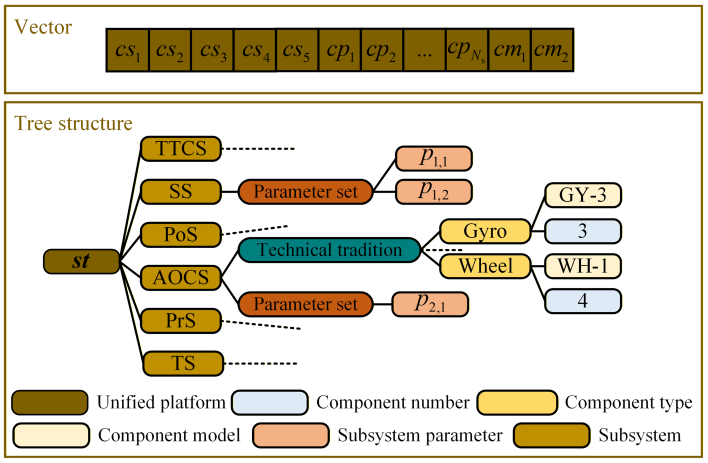

Figure 4: Genetic encoding of individuals.

Table 2: Optimization focuses selected for different stages.

Genetic Codes of the Individual	First Stage		Second Stage	
	Outer Loop	Inner Loop	Outer Loop	Inner Loop
cs_1, cs_2, cs_3, cs_4, cs_5, cp_i $(1 \le i \le N_s)$	Changeable	Unchangeable	Unchangeable	Unchangeable
st	Unchangeable	Unchangeable	Changeable	Unchangeable
cm_1, cm_2	Unchangeable	Changeable	Unchangeable	Changeable

3.2 Initialize Parent Population

N_{pop} individuals at the first stages are randomly generated in the initial stages to form the parent population, based on the problem encoding described in Section 3.1.

3.3 Adaptive Evolutionary Operators to Create Offspring Population

In traditional intelligent algorithms, crossover and mutation probabilities are typically preset constants that ignore the varying influence of different optimization variables on objective values. This section proposes adaptive evolutionary operators based on online contribution learning to address this limitation. The core principle is to enable TNC-A to learn the historical contribution of each optimization variable during the optimization process. Consequently, search resources are dynamically and differentially allocated, thereby focusing computational effort on the most critical optimization variables.

The process described in this section is outlined in Algorithm 1.

Algorithm 1: The adaptive evolutionary operators to create offspring population

Input: Parent population $\boldsymbol{PaP}$, N_C^{RE}, N_M^{RE}, $\boldsymbol{OCT}$
Output: Offspring population $\boldsymbol{OfP}$, evolution log $\boldsymbol{EL}$
1: Create two counter variables: $N_C^{CU} = 0$, $N_M^{CU} = 0$.
2: **while** $N_C^{CU} < N_C^{RE}$ **do**
3: Randomly select two individuals at the same stage from $\boldsymbol{PaP}$: $\boldsymbol{In}_i = \{\boldsymbol{Ve}_i, \boldsymbol{St}_i\}$,$\boldsymbol{In}_j = \{\boldsymbol{Ve}_j, \boldsymbol{St}_j\}$.

(Continued)

Algorithm 1 (continued)

4: Perform the adaptive vector crossover operator between $\boldsymbol{Ve}_i$ and $\boldsymbol{Ve}_j$ to create $\boldsymbol{Ve}_l$ and $\boldsymbol{Ve}_m$.
5: Perform the adaptive tree-structured crossover operator between $\boldsymbol{St}_i$ and $\boldsymbol{St}_j$ to create $\boldsymbol{St}_l$ and $\boldsymbol{St}_m$.
6: Create two new individuals: $\boldsymbol{In}_l = \{\boldsymbol{Ve}_l, \boldsymbol{St}_l\}$, $\boldsymbol{In}_m = \{\boldsymbol{Ve}_m, \boldsymbol{St}_m\}$.
7: $N_{\mathrm{C}}^{\mathrm{CU}} = N_{\mathrm{C}}^{\mathrm{CU}} + 2$.
8: **end while**
9: **while** $N_{\mathrm{M}}^{\mathrm{CU}} < N_{\mathrm{M}}^{\mathrm{RE}}$ **do**
10: Randomly select one individual from $\boldsymbol{PaP}$: $\boldsymbol{In}_k = \{\boldsymbol{Ve}_k, \boldsymbol{St}_k\}$.
11: Perform the adaptive vector mutation operator between $\boldsymbol{Ve}_k$ to create $\boldsymbol{Ve}_n$.
12: Perform the adaptive tree-structured mutation operator between $\boldsymbol{St}_k$ to create $\boldsymbol{St}_n$.
13: Create one new individual: $\boldsymbol{In}_n = \{\boldsymbol{Ve}_n, \boldsymbol{St}_n\}$.
14: $N_{\mathrm{M}}^{\mathrm{CU}} = N_{\mathrm{M}}^{\mathrm{CU}} + 1$.
15: **end while**
16: $\boldsymbol{OfP}$ is composed of $N_{\mathrm{C}}^{\mathrm{RE}} + N_{\mathrm{M}}^{\mathrm{RE}}$ new created individuals.

In Algorithm 1, $\boldsymbol{In}_i$ represents the i-th individual, which comprises two parts: a $(N_{\mathrm{s}} + 5)$-dimensional vector $\boldsymbol{Ve}_i$ (corresponding to the vector without the optimization variables cm_1 and cm_2) and a tree structure $\boldsymbol{St}_i$ (corresponding to the optimization variable $\boldsymbol{st}$). The online contribution tables $\boldsymbol{OCT} = \{H_1, H_2, \ldots, H_{18+N_{\mathrm{SP}}+3N_{\mathrm{Com}}}\}$ record the respective contribution values of each vector part and tree node, as mentioned in Section 3.4. $\boldsymbol{EL}$ records the execution logs of $N_{\mathrm{C}}^{\mathrm{RE}}$ adaptive crossover operators and $N_{\mathrm{M}}^{\mathrm{RE}}$ adaptive mutation operators, as detailed in Section 3.4.

The procedure for the adaptive vector crossover operator described in Algorithm 1 is presented in Algorithm 2.

Algorithm 2: The adaptive vector crossover operator

Input: $\boldsymbol{Ve}_i$, $\boldsymbol{Ve}_j$, $\boldsymbol{OCT}$, $\boldsymbol{EL}$
Output: $\boldsymbol{Ve}_l$, $\boldsymbol{Ve}_m$, $\boldsymbol{EL}$

1: Create a random decimal: $rn_1 \in [0, 1]$.
2: **if** $rn_1 \leq P_{\mathrm{vc}}$ **then**
3: Define the optional set $\boldsymbol{OS} = \{1, 2, \ldots, 6\}$.
4: Create a random real number: $rn_2 \in [1, 6]$.
5: **for** $k = 1$ to rn_2 **do**
6: Calculate the selection probability for each part in the $\boldsymbol{OS}$.
7: Select one part through the weighted roulette-wheel selection based on its probability.
8: Remove the index of the selected part from the $\boldsymbol{OS}$.
9: **end for**
10: Select crossover points based on selected rn_2 parts.
11: Perform the standard multi-point crossover between $\boldsymbol{Ve}_i$ and $\boldsymbol{Ve}_j$ at the selected crossover points to create $\boldsymbol{Ve}_l$ and $\boldsymbol{Ve}_m$.
12: **else if**
13: $\boldsymbol{Ve}_l = \boldsymbol{Ve}_i$ and $\boldsymbol{Ve}_m = \boldsymbol{Ve}_j$.
14: **end if**
15: Update $\boldsymbol{EL}$.

In Step 3 of Algorithm 2, the elements in $\boldsymbol{OS}$ correspond to 6 different parts of the vector, which are cs_1, cs_2, cs_3,cs_4, cs_5 and $\{cp_1, cp_2, \ldots, cp_{N_s}\}$, respectively.

In Step 6 of Algorithm 2, the selection probability for each part is calculated as

$$P_n^{\mathrm{Sel}} = \begin{cases} H_n / \sum\limits_{x \in \boldsymbol{OS}} H_x, & n-\text{th part} \in \boldsymbol{OS} \\ 0, & n-\text{th part} \notin \boldsymbol{OS} \end{cases} \tag{9}$$

where P_i^{Sel} is the selection probability of the n-th part.

In Step 10 of Algorithm 2, if any of the parts cs_1, cs_2, cs_3, cs_4 and cs_5 is selected, the crossover point will be itself. If the part $\{cp_1, cp_2, \ldots, cp_{N_s}\}$ is selected, rn_3 $(1 \le rn_3 \le N_s)$ random points within the $\{cp_1, cp_2, \ldots, cp_{N_s}\}$ are designated as crossover points.

The procedure for the adaptive vector mutation operator closely resembles that for the adaptive vector crossover operator. Given a preset probability P_{vm}, standard multi-point mutation is employed.

As shown in Fig. 5, six distinct tree-structured crossover methods are applied to the true structure: subsystem (S), subsystem parameter set (SPS), subsystem parameter (SP), component category (CC), component model (CM), and component number (CN) crossovers. The number of nodes available for crossover varies by method. S, SPS, and SP crossovers involve 6 nodes. SP crossover involves N_{SP} nodes, where N_{SP} is the total number of subsystem parameters within a satellite. CC, CM and CN crossover involves N_{Com} nodes, where N_{Com} is the total number of component types within a satellite.

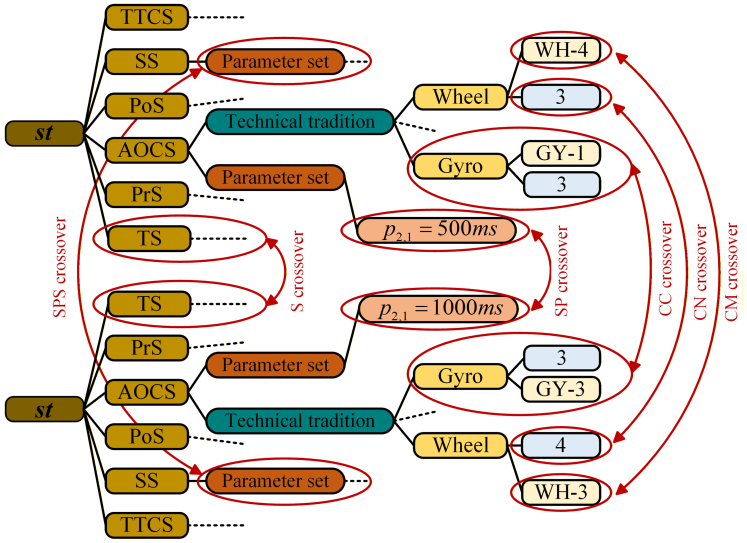

Figure 5: Six different crossover methods for the tree structure.

The process of the adaptive tree-structured crossover operator is illustrated in Algorithm 3.

Algorithm 3: The adaptive tree-structured crossover operator

Input: $\boldsymbol{St}_i$, $\boldsymbol{St}_j$, $\boldsymbol{OCT}$, $\boldsymbol{EL}$

Output: $\boldsymbol{St}_l$, $\boldsymbol{St}_m$, $\boldsymbol{EL}$

1: Create a random decimal: $rn_1 \in [0,1]$.

2: **if** $rn_1 \le P_{\mathrm{sc}}$ **then**

(Continued)

Algorithm 3 (continued)

Create a random real number: $rn_2 \in [1, 6]$, and randomly select rn_2 tree-structured crossover methods.
for each selected tree-structured crossover method **do**
  Define the optional set $\boldsymbol{OS}$ based on the selected tree-structured crossover method.
  Create a random real number: $rn_3 \in [1, N_{OS}]$, and N_{OS} is the total number of elements in the $\boldsymbol{OS}$.
  for $k = 1$ to rn_3 **do**
    Calculate the selection probability for each node in the $\boldsymbol{OS}$ using Eq. (9).
    Select one node through the weighted roulette-wheel selection based on its probability.
    Remove the index of the selected node from the $\boldsymbol{OS}$.
  end for
end for
Perform an exchange between $\boldsymbol{St}_i$ and $\boldsymbol{St}_j$ at all selected nodes to create $\boldsymbol{St}_l$ and $\boldsymbol{St}_m$.
else if
  $\boldsymbol{St}_l = \boldsymbol{St}_i$ and $\boldsymbol{St}_m = \boldsymbol{St}_j$.
end if
Update $\boldsymbol{EL}$.

In Step 5 of Algorithm 3, offspring set ($\boldsymbol{OS}$) can be expressed as:

$$\boldsymbol{OS} = \begin{cases} \{7, 8, \ldots, 12\} & \text{, Scrossover is selected} \\ \{13, 14, \ldots, 18\} & \text{, SPS crossover is selected} \\ \{19, 20, \ldots, 18 + N_{SP}\} & \text{, SP crossover is selected} \\ \{19 + N_{SP}, 20 + N_{SP}, \ldots, 18 + N_{SP} + N_{Com}\} & \text{, CC crossover is selected} \\ \{19 + N_{SP} + N_{Com}, 20 + N_{SP} + N_{Com}, \ldots, 18 + N_{SP} + 2N_{Com}\} & \text{, CM crossover is selected} \\ \{19 + N_{SP} + 2N_{Com}, 20 + N_{SP} + 2N_{Com}, \ldots, 18 + N_{SP} + 3N_{Com}\} & \text{, CN crossover is selected} \end{cases} \tag{10}$$

where $\boldsymbol{OS}$ varies according to the selected tree-structured crossover method. The elements in the $\boldsymbol{OS}$ correspond to all nodes available for crossover in the selected tree structure method, as mentioned in Section 3.4.

The applied adaptive tree-structured mutation operator $\boldsymbol{St}_k$ can be regarded as an adaptive tree-structured crossover between $\boldsymbol{St}_k$ and a randomly generated tree structure. The preset probability P_{sm} is specified.

3.4 Online Contribution Tables

The online contribution tables $\boldsymbol{OCT}$ in Fig. 6 record the contribution values of the six vector parts and $3N_{Com} + N_{SP} + 12$ nodes in the tree structure.

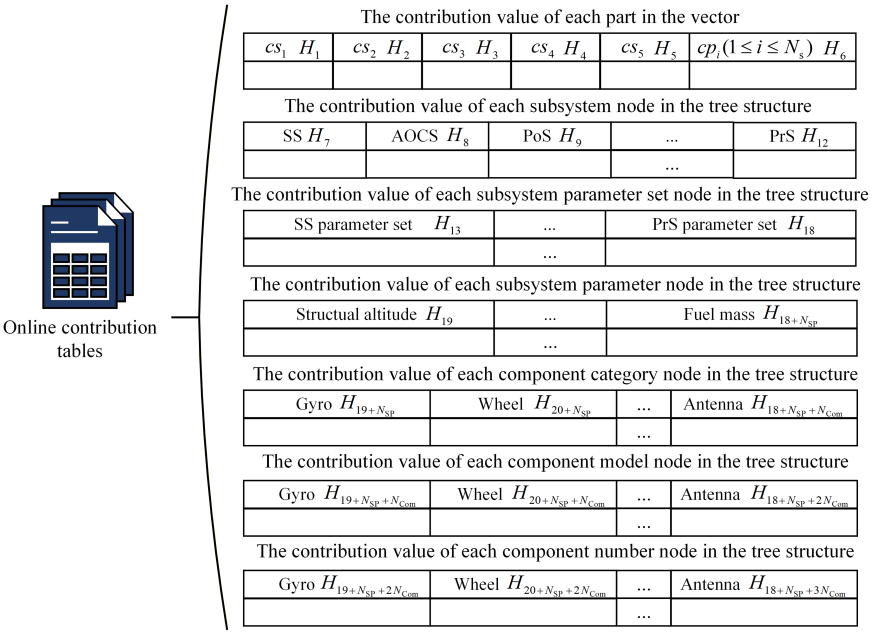

Figure 6: Online contribution tables.

The contribution update process of $\boldsymbol{OCT}$ is shown in Algorithm 4.

Algorithm 4: The contribution update of the online contribution tables

Input: Evolution log $\boldsymbol{EL}$, fitness value of each individual within the combined population, $\boldsymbol{OCT}$
Output: updated $\boldsymbol{OCT}$

for each $\boldsymbol{el}_i$ in $\boldsymbol{EL}$ **do**
  if $type_i$ is crossover **then**
    $\Delta fit = (\max(0, (fit(opi_i) + fit(api_i) - fit(ooi_i) - fit(aoi_i)))) \cdot (2 - fit(opi_i) - fit(api_i))/4$
  else if
    $\Delta fit = (\max(0, (fit(opi_i) - fit(ooi_i)))) \cdot (1 - fit(opi_i))$
  end if
  for each $dn_{i,j}$ in $\boldsymbol{dn}_i$ **do**
    if $dn_{i,j} = 1$ **then**
      Update the contribution value of the j-th part or node.
    end if
  end for
end for

In Algorithm 4, $\boldsymbol{EL}$ records the execution logs of N_C^{RE} adaptive crossover operators and N_M^{RE} adaptive mutation operators mentioned in Algorithm 1, as seen in Table 3. In Table 3, the i-th execution log $\boldsymbol{el}_i$ can be defined as $\boldsymbol{el}_i = \{type_i, opi_i, api_i, ooi_i, aoi_i, \boldsymbol{dn}_i\}$. $type_i$ is the adaptive operator type of the i-th execution log. opi_i and api_i represent indices of the two parent individuals within the combined population. ooi_i and aoi_i represent indices of the two offspring individuals within the combined population. $\boldsymbol{dn}_i = [dn_{i,1}, dn_{i,1}, \ldots, dn_{i,3N_{Com}+N_{SP}+18}]$ is a binary vector, and all the elements in the $\boldsymbol{dn}_i$ correspond to all parts and nodes shown in Fig. 6. Specifically, the dimensions 1~6 of $\boldsymbol{dn}_i$ correspond to 6 parts of the vector, which

are $cs_1, cs_2, cs_3, cs_4\ cs_5$ and $\{cp_1, cp_2, \ldots, cp_{N_s}\}$, respectively. The dimensions 7~12, 13~18 and 19 ~ $(N_{SP} + 18)$ of $\boldsymbol{dn}_i$ correspond to 6 subsystem nodes, 6 subsystem parameter set nodes, and N_{SP} subsystem parameter nodes, respectively. The dimensions $(N_{SP} + 19) \sim (N_{Com} + N_{SP} + 18)$ of $\boldsymbol{dn}_i$ correspond to N_{Com} component category nodes. The dimensions $(N_{SP} + N_{Com} + 19) \sim (2N_{Com} + N_{SP} + 18)$ of $\boldsymbol{dn}_i$ correspond to the N_{Com} component model nodes. The dimensions $(N_{SP} + 2N_{Com} + 19) \sim (3N_{Com} + N_{SP} + 18)$ of $\boldsymbol{dn}_i$ correspond to N_{Com} component number nodes. For each dimension of $\boldsymbol{dn}_i$, a value of 1 indicates that the corresponding part or node has been selected for operator execution.

Table 3: Example of execution logs.

Execution Log Index	1	...	41	...
Adaptive operator type $type_i$	Crossover	...	Mutation	...
The index of one parent individual opi_i	29	...	37	...
The index of another parent individual api_i	7	...	None	...
The index of one offspring individual ooi_i	61	...	101	...
The index of another offspring individual aoi_i	62	...	None	...
All dimensions and nodes selected for operator execution $\boldsymbol{dn}_i$	...	...	...	...

In Step 9 of Algorithm 4, the contribution value of the j-th part or node is calculated as

$$H_j = (1 - \alpha) H_j + \alpha \cdot \Delta fit, \tag{11}$$

where H_j is the contribution value of the j-th part or node and α is the preset learning rate.

3.5 Combination of Parent and Offspring Populations

The combined population, comprising individuals, is formed by merging the parent and offspring populations, which comprises $N_{pop} + N_C^{RE} + N_M^{RE}$ individuals.

3.6 Automatic Modeling, Simulation, and Objective Calculation

This section details the inner loop optimization, which is executed for each individual within the combined population generated by the outer loop, as illustrated in Algorithm 5. Given the morphology of each individual, the optimal mission planning parameters cm_1 and cm_2 are determined through inner loop optimization. As described in Section 2.1, both cm_1 and cm_2 assume values from the discrete set {0.1, 0.2, ..., 1.0}, and thus there are 100 different (cm_1, cm_2) combinations.

Algorithm 5: The automatic modeling simulation and objective calculation

Input: The combined population ***CoP***
Output: Fitness value of each individual within the combined population

```
for each individual In_i in the CoP then
    Construct the MWC model for the In_i
    for cm_1 = 0.1 to 1 by 0.1 do
        for cm_2 = 0.1 to 1 by 0.1 do
            Configure the (cm_1, cm_2) for the MWC model of In_i to create one condition.
        end for
    end for
```

(Continued)

Algorithm 5 (continued)

8: **end for**
9: Simulate 100 conditions of each individual in the ***CoP*** to calculate their own $F(\boldsymbol{X})$.
10: **for** each individual $\boldsymbol{In}_i$ in the ***CoP*** **then**
11: $fit(\boldsymbol{In}_i) = \max\left\{F(\boldsymbol{X})_{i,1}, F(\boldsymbol{X})_{i,2}, \ldots, F(\boldsymbol{X})_{i,100}\right\}$.
12: Update the $\boldsymbol{In}_i$ with the (cm_1, cm_2) corresponding to the condition that achieves the maximum $F(\boldsymbol{X})$.
13: **end for**

In Step 11 of Algorithm 5, $fit(\boldsymbol{In}_i)$ is the fitness value of the i-th individual, and $F(\boldsymbol{X})_{i,1}$ is the performance value $F(\boldsymbol{X})$ calculated by simulating the j-th condition of the i-th individual.

The fitness value for all individuals in the combined population are determined.

3.7 Selection Operator to Form a New Parent Population

The selection operator is applied to the combined population. First, all individuals are sorted in ascending order of their fitness value, with lower values indicating superior performance. The top N_{pop} individuals with the lowest fitness values are then selected as the new parent population.

3.8 Individual Stage Switching

Stage switching is restricted to individuals at the first stage. These individuals are promoted to the second stage if they survive the selection operator and maintain a fitness value that is superior to the population average for N_{SSI} consecutive iterations. Individuals at the second stage do not undergo stage switching.

4 Results and Discussion

The simulation results presented in Section 4.1 were obtained using 50 computers, each equipped with an 8-core i7-9700k CPU and 16 GB of RAM. In Section 4.2, the computational resources were extended to include these computers and the high-performance computing platform of Beihang University.

The MWC model parameters are defined as follows. The defensive force comprises five military bases and one missile warning constellation. Each military base is fixed geographically and prepositioned with 10 interception missiles. The offensive force launches four missiles against each defensive military base. Both ω_1 and ω_2 are set to 0.5. The simulation starts at 00:00:00 on 01 January 2025, with a step size of 1 s. The simulation terminates once all offensive missiles have either been intercepted or reached their destinations.

Table 4 lists the TNC-A parameters.

Table 4: The pre-set parameters in the TNC-A.

Parameters	Values	Parameters	Values	Parameters	Values
N_{MaxI}	150	N_{SSI}	10	N_{pop}	60
N_{C}^{RE}	40	N_{M}^{RE}	20	P_{vc}	0.7
P_{sc}	0.7	P_{vm}	0.3	P_{sm}	0.3
α	0.2	N_{s}	300		

Tables 5–7 list the feasible ranges for all optimization variables.

Table 5: Feasible ranges of all optimization variables.

Optimization Variables	Ranges	Optimization Variables	Ranges
cs_1/	[5, 20]	cs_5/	$[0, cs_1 - 1]$
cs_2/	[5, 30]	cp_i $(1 \le i \le N_{\text{sat}})$	[1, 12] ($N_{\text{pay}} = 12$)
cs_3/°	[60, 100]	***st***	See Tables 6 and 7
cs_4/km	[400, 1000]	cm_1, cm_2	$\{0.1, 0.2, \ldots, 1.0\}$ for each

Table 6: Optional ranges of selected typical components within ***st***.

Components	Optional Ranges
Gyro	Optional models: GY-1, GY-2, GY-3, GY-4, GY-5 Optional number: 3, 4, 5, 6
Star sensor	Optional models: ST-1, ST-2, ST-3, ST-4, ST-5 Optional number: 1, 2
Wheel	Optional models: WH-1, WH-2, WH-3, WH-4, WH-5 Optional number: 3, 4, 5, 6
Battery	Optional models: BA-1, BA-2, BA-3, BA-4, BA-5 Optional number: 2, 4
Antenna	Optional models: AN-1, AN-2, AN-3, AN-4, AN-5 Optional number: 1, 2
Fuel tank	Optional models: FU-1, FU-2, FU-3, FU-4 Optional number: 1, 2

Table 7: Typical ranges for selected subsystem parameters in ***st***.

Subsystem Parameters	Optional Ranges	Subsystem Parameters	Optional Ranges
Cross-sectional envelope radius/m	[0.2, 2]	Heat dissipation window area/m^2	[0.1, 1]
Control cycle/ms	[500, 2000]	Radio frequency carrier frequency/GHz	[1, 10]
Solar array area/m^2	[2, 10]	Fuel mass/kg	[20, 200]

The optional range cp_i is [1,12], corresponding to 12 distinct camera models; the first six are wide-field models, whereas the last six are high-resolution models.

4.1 Effectiveness Analysis of the TNC-A

This section evaluates the effectiveness of TNC-A.

Fig. 7 illustrates the iteration process of TNC-A. After 101 iterations, the algorithm gradually converged toward the optimal fitness value of 0.460, which was maintained until the 150th iteration. The total computational time for this process was 15.51 days.

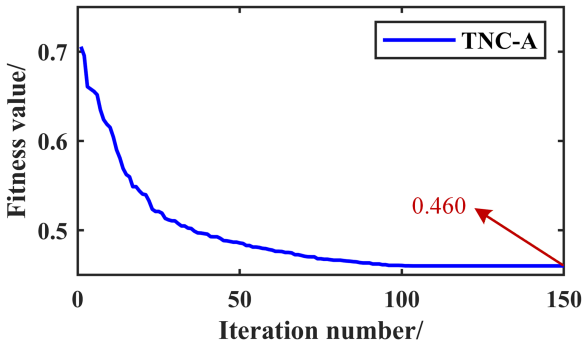

Figure 7: TNC-A iteration process.

During the TNC-A iteration process, the online contribution tables were continuously updated, causing the contribution values of each dimension or node to vary. The wheel, battery, and antenna component model nodes within the tree structure serve as examples of these variations (Fig. 8). All three curves exhibit a consistent three-phase pattern, reflecting the interaction between the online contribution learning mechanism and the search process. In the initial phase (approximately iterations 1–11), no second-stage individuals were generated. TNC-A did not perform adaptive tree-structured crossover or mutation, and the contribution values of these three nodes remained at 0. In the second phase (approximately iterations 12–35), the number of second-stage individuals increased, prompting TNC-A to conduct a broad search across these three nodes in the high-dimensional tree structure. The adaptive tree-structured crossover and mutation operators frequently modify key component models, producing significant $\Delta fit > 0$, albeit occasionally adverse $\Delta fit < 0$, performance feedback. The contribution values of these three nodes underwent significant oscillations. In the third phase (approximately 36–150 iterations), as the population converged toward promising regions of the solution space, modifications to these key component models yielded diminishing returns (Δfit gradually decreased toward 0). Consequently, the oscillations for these three nodes were rapidly dampened before gradually converging and stabilizing at near-zero levels.

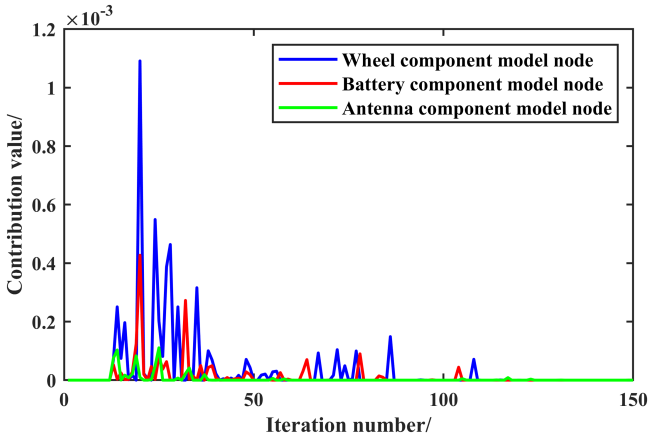

Figure 8: Variation in the contribution values of the three component model nodes during the TNC-A iteration process.

Furthermore, Fig. 8 illustrates disparities in the oscillation amplitudes of different nodes, which can be attributed to their differing criticality and direct influence on performance $F(\boldsymbol{X})$. In the MWC model, the wheel serves as a critical actuator for attitude control. Any change to its model directly influences the entire attitude control process, thereby having an immediate and significant impact on core mission functions (specifically, the identification and tracking of incoming missiles). This direct coupling results in substantial performance feedback Δfit during the TNC-A iteration process. Conversely, the battery and antenna impacts are more indirect and moderated. The battery primarily constrains mission execution by limiting the operational availability of the satellite, which indirectly influences performance $F(\boldsymbol{X})$. The antenna primarily affects the data transmission capability of the satellite, which is not the primary bottleneck in performance $F(\boldsymbol{X})$. Accordingly, the contribution value of the wheel exhibits larger updates and more pronounced oscillations than those of the battery and antenna.

The optimal scheme was obtained after 150 iterations, and the values of its partial typical optimization variables are illustrated in Fig. 9. This scheme can be directly used to generate a comprehensive mission warning constellation design report, guiding the manufacture, launch, and operation of each satellite. These results demonstrate the significant practical potential of TNC-A for engineering applications.

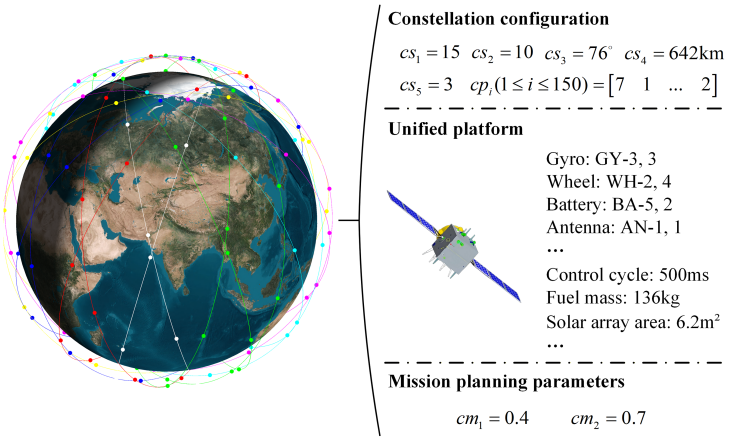

Figure 9: Partial typical optimization variables of the optimal scheme obtained by the TNC-A.

4.2 Performance Analysis of the TNC-A

To conduct an in-depth performance analysis of the TNC-A, a comprehensive comparative analysis was performed against three representative algorithms, selected according to ablation study principles. Specifically, ASAGA [28] serves as the baseline for advanced intelligent algorithms, whereas TSO serves as the baseline for traditional engineering methods. TSO adopts a multi-stage optimization strategy combined with a sequential optimization structure, providing a feasible pathway for the preliminary alignment between component-level constellation morphology and mission planning. However, TSO inherently suffers from evaluation distortion due to its sequential optimization structure. To address this limitation, TNC evolves from TSO by replacing the sequential optimization structure with a double-nested optimization structure, thereby mitigating evaluation distortion. Finally, TNC-A incorporates adaptive evolutionary operators, allowing for the explicit isolation of the contribution of these operators through comparison with TNC. For a fair comparison, the multi-stage optimization process for TSO and TNC (including stage division and switching) was kept consistent with that of TNC-A. Both TNC and TSO adopted fixed-probability operators, which implicitly assume that the contribution values of all nodes and parts mentioned in Fig. 6 are equal.

Furthermore, TNC-A imposes a heavy computational burden due to its unique double-nested optimization structure. In practical engineering, optimization tasks are invariably constrained by limited

computational budgets. Hence, based on Reference [29], this section compares TNC-A with other algorithms under a limited budget of function evaluations.

The parameters of all algorithms are described as follows. The population size in ASAGA was 60, and other parameters were set based on Reference [28]. Note that because ASAGA did not incorporate a tree-structured representation, the optimization variable ***st***, which was encoded as a tree structure, was transformed into a real-valued vector. In TSO, both cm_1 and cm_2 were fixed at 0.5 when optimizing constellation morphology. The population size was 60 for TSO, TNC, and TNC-A. Except for ASAGA, these four algorithms share the same values of N_C^{RE}, N_M^{RE}, P_{vc}, P_{sc}, P_{vm} and P_{sm}, which are given in Table 4. Given a limited budget of function evaluations, the total number of function evaluations was uniformly set to 150,000 for each algorithm. All algorithms were run 10 times to ensure fairness in the performance comparison. Table 8 compares the maximum (Max), minimum (Min), average (Avg), and standard deviation (Std) of the optimization results. Furthermore, the statistical significance of the performance differences was assessed using the two-sided Wilcoxon rank-sum test at a significance level of $\alpha = 0.05$. This non-parametric test is particularly suitable for small sample sizes because it does not assume a normal distribution and relies solely on the rank ordering of the data. For each algorithm pair, the null hypothesis (H_0) states that the median performance values are equal. The test produces a p-value, and h is a binary decision variable. If the p-value is less than 0.05, h = 1, leading to the rejection of H_0; otherwise, $h = 0$, indicating a failure to reject H_0 and suggesting no significant difference between the two algorithms. Table 9 presents the relevant results of the two-sided Wilcoxon rank-sum test.

Table 8: Statistical comparison of optimization results for different algorithms.

Different Algorithms	Optimal Fitness Value			
	Max	**Min**	**Avg**	**Std**
ASAGA	0.7347	0.6412	0.6943	0.0309
TSO	0.6741	0.5250	0.6226	0.0504
TNC	0.6015	0.5588	0.5836	0.0128
TNC-A	0.5676	0.5111	0.5374	0.0175

Table 9: Results of the two-sided Wilcoxon rank-sum test.

Algorithm Pairs	*p*-Value	*h*
TNC-A vs. TNC	3.3E−4	1
TNC-A vs. TSO	0.0013	1
TNC-A vs. ASAGA	1.8E−4	1
TNC vs. TSO	0.0452	1
TSO vs. ASAGA	0.0036	1

In Table 8, TNC-A achieved the best overall performance, followed by TNC, TSO, and ASAGA. In Table 9, for all algorithm pairs, p-values below 0.05 and $h = 1$ indicate rejection of the null hypothesis of equal medians (two-sided test) at the 5% significance level.

Compared with ASAGA, TSO demonstrates substantially improved performance on the large-scale co-optimization problem of constellation morphology and mission planning. This improvement stems from the sequential optimization structure, which initially decouples the co-optimization problem into

two independent subproblems, followed by multi-stage optimization that reduces the complexity of the subproblem. This comparison suggests that advanced intelligent algorithms, when applied directly without a purpose-built framework, fail to effectively address the co-optimization problem investigated in this study. However, the sequential optimization strategy in TSO suffers from evaluation distortion, which ultimately manifests as reduced stability in the optimization results.

Compared with TSO, TNC exhibits superior overall performance, particularly in terms of stability, as indicated by its markedly smaller standard deviation. This improvement stems from the TNC's double-nested optimization structure, which addresses the evaluation distortion inherent in TSO. Individuals are evaluated in the sequential optimization structure of TSO without optimizing their corresponding mission planning parameters, leading to inaccurate fitness values (Fig. 10). Five schemes (S1–S5) were randomly selected from the parent population generated at 60,000 function evaluations during a TNC run. The mission planning coefficients cm_1 and cm_2 for these five schemes were fixed at 0.5 to simulate the fixed mission planning evaluation of TSO. The fitness values of these modified schemes were re-evaluated and compared with the original fitness values obtained under TNC inner-loop optimization. The results demonstrate a consistent deterioration in fitness under fixed mission planning parameters, confirming that the evaluation distortion in the sequential optimization structure underestimates the scheme's true potential. Conversely, the double-nested optimization structure of TNC evaluates each scheme under its optimal mission planning parameters, thereby avoiding this pitfall.

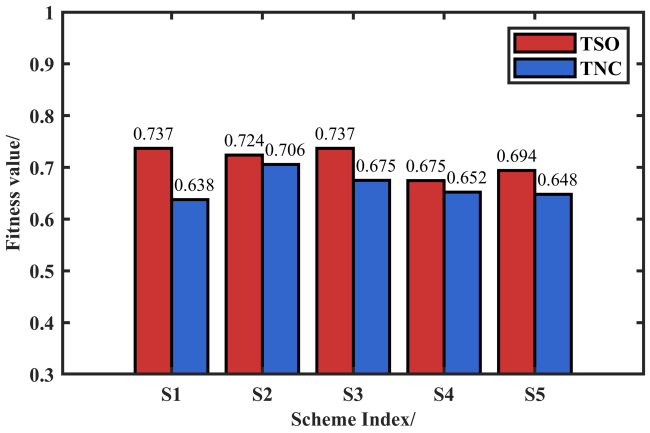

Figure 10: Impact of different algorithms (TSO and TNC) on the fitness values of 5 schemes.

TNC-A demonstrates superior overall performance compared with TNC due to its adaptive evolutionary operators, which dynamically adjust the search focus to maximize search efficiency (see Fig. 11).

Fig. 11 illustrates the convergence curves of TNC and TNC-A. Both algorithms initially exhibit similar improvement rates because the online contribution tables in TNC-A have not yet accumulated sufficient meaningful data. However, after approximately 40,000 function evaluations, TNC-A began to outperform TNC, demonstrating a noticeably accelerated improvement in the optimal fitness value. This acceleration stems from adaptive evolutionary operators that leverage historical experience to identify promising vector components and tree structure nodes, thereby assigning them higher selection probabilities. Consequently, TNC-A allocates computational resources more efficiently to achieve rapid convergence than TNC, which relies on uniform random search and expends effort on less impactful regions.

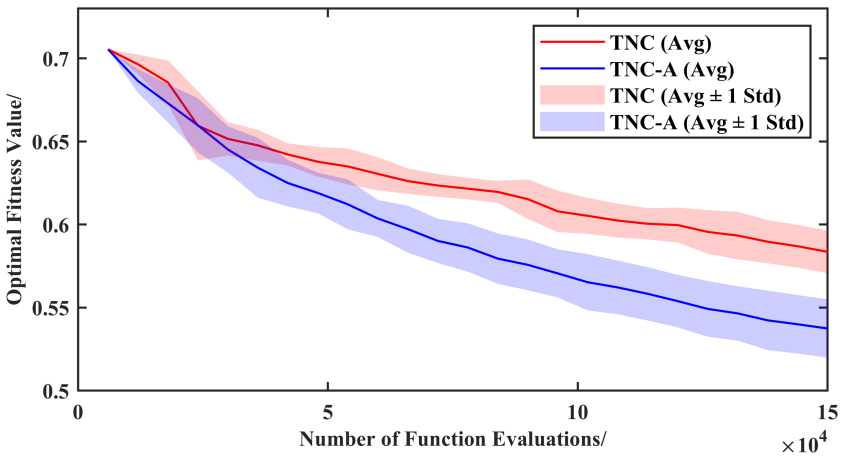

Figure 11: Convergence comparison of different algorithms (TNC-A and TNC).

5 Conclusions

This study addresses the key challenges of evaluation distortion, the curse of dimensionality, and inefficient search in the co-optimization of component-level missile warning constellation morphology and mission planning. A multi-stage optimization strategy, combined with a double-nested optimization structure, is designed to alleviate the curse of dimensionality and overcome the evaluation distortion inherent in traditional sequential optimization structures. In addition, adaptive evolutionary operators based on online contribution learning are employed to enhance search efficiency. The simulation results confirm the effectiveness and practical viability of TNC-A, demonstrating its superiority over existing algorithms under a limited budget of function evaluations. Specifically, with a budget of 150,000 function evaluations, the TNC-A outperformed ASAGA, achieving approximately a 22.6% reduction in the average value and a 43.4% reduction in the standard deviation of the optimization results. Furthermore, compared with TSO, TNC-A achieved reductions of 13.7% and 65.3% in the average and standard deviation, respectively.

The TNC-A will be further investigated and extended in future work. Promising research directions include, but are not limited to, exploring alternative mission-planning strategies, conducting comprehensive comparisons with a broader range of state-of-the-art baseline algorithms, extending the current single-objective formulation to a Pareto-based multi-objective optimization framework, and performing systematic scalability analyses under increasingly complex problem settings.

Acknowledgement: Not applicable.

Funding Statement: This work was supported in part by the Key Laboratory of Spacecraft Design Optimization and Dynamic Simulation Technologies, Ministry of Education.

Author Contributions: The authors confirm contribution to the paper as follows: Conceptualization, Yunfeng Dong; methodology, Chao Zhang and Yunfeng Dong; software, Chao Zhang; validation, Chao Zhang and Yunfeng Dong; formal analysis, Chao Zhang; investigation, Chao Zhang; resources, Chao Zhang; data curation, Chao Zhang and Yunfeng Dong; writing—original draft preparation, Chao Zhang; writing—review and editing, Chao Zhang and Yunfeng Dong; visualization, Chao Zhang; supervision, Yunfeng Dong; project administration, Chao Zhang; funding acquisition, Yunfeng Dong. All authors reviewed and approved the final version of the manuscript.

Availability of Data and Materials: The data that support the findings of this study are available from the Corresponding Author, [YD], upon reasonable request.

Ethics Approval: Not applicable.

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

References

1. Qin Z, Liang YG. Sensor management of LEO constellation based on covariance control. J Syst Eng Electron. 2019;30(2):393–401. doi:10.21629/JSEE.2019.02.17.
2. Li J, Ye D, Xiao Y, Zhang Z. Distributed task scheduling method for low earth orbit early warning constellation. Sci Sin-Phys Mech Astron. 2025;55(9):294507. doi:10.1360/sspma-2024-0493.
3. Budianto IA, Olds JR. Design and deployment of a satellite constellation using collaborative optimization. J Spacecr Rockets. 2004;41(6):956–63. doi:10.2514/1.14254.
4. Choo N, Ahner D, Little B. A survey of orbit design and selection methodologies. J Astronaut Sci. 2024;71(1):4. doi:10.1007/s40295-023-00420-9.
5. Carden J, Deacon S, Kessler P, Speth P. Generation-based evolutionary tool for the optimization of constellations (GenETOC). In: Proceedings of the 2021 IEEE Aerospace Conference; 2021 Mar 6–13; Big Sky, MT, USA. Piscataway, NJ, USA: IEEE; 2021. p. 1–11. doi:10.1109/aero50100.2021.9438213.
6. Qin C, Gao Y, Wang Y. The optimization of low earth orbit satellite constellation visibility with genetic algorithm for improved navigation potential. Sci Rep. 2025;15(1):30798. doi:10.1038/s41598-025-16815-7.
7. Li P, Wang H, Zhang Y, Pan R. Mission planning for distributed multiple agile Earth observing satellites by attention-based deep reinforcement learning method. Adv Space Res. 2024;74(5):2388–404. doi:10.1016/j.asr.2024.06.003.
8. Qin J, Li B, Bai X, Ran D, Xu M, Zhang R, et al. Distributed autonomous scheduling based on event trigger for heterogeneous satellite swarm. Chin Space Sci Technol. 2025;45(4):88–101. (In Chinese). doi:10.16708/j.cnki.1000-758X.2025.0061.
9. Yang Y, Liu D. Distributed imaging satellite mission planning based on multi-agent. IEEE Access. 2023;11:65530–45. doi:10.1109/ACCESS.2023.3289964.
10. Li X, Chen Y, Xing L, Chen Y, Du Y, He L. A review of the frameworks, models, and algorithms for large-scale imaging satellite mission planning. Expert Syst Appl. 2025;292(1):128471. doi:10.1016/j.eswa.2025.128471.
11. Shang M, Yuan R, Song B, Huang X, Yang B, Li S. Joint observation and transmission scheduling of multiple agile satellites with energy constraint using improved ACO algorithm. Acta Astronaut. 2025;230(12):92–103. doi:10.1016/j.actaastro.2025.02.008.
12. Guo M, Wu X, Luo Z, Song X. An improvement of contract net protocol for distributed satellite collaborative task planning. Int J Aerosp Eng. 2026;2026(1):5536856. doi:10.1155/ijae/5536856.
13. Al-Helali B, Chen Q, Xue B, Zhang M. GP with a hybrid tree-vector representation for instance selection and symbolic regression on incomplete data. In: Proceedings of the 2021 IEEE Congress on Evolutionary Computation (CEC); 2021 Jun 28–Jul 1; Kraków, Poland. Piscataway, NJ, USA: IEEE; 2021. p. 604–11. doi:10.1109/cec45853.2021.9504767.
14. Xie S, Li Z, Dong Y, Chen P. Multigranularity batch sequential design method for component-level satellite design optimization. IEEE Trans Aerosp Electron Syst. 2024;60(6):8779–90. doi:10.1109/TAES.2024.3433325.
15. Gu X, Zeng Y, Ga L, Gao Y. High-efficiency design of mega-constellation based on genetic algorithm coverage optimization. Symmetry. 2025;17(10):1619. doi:10.3390/sym17101619.
16. Chatterjee A, Tharmarasa R. Multi-stage optimization framework of satellite scheduling for large areas of interest. Adv Space Res. 2024;73(3):2024–39. doi:10.1016/j.asr.2023.11.016.
17. Kwon J, Kim S, Park FC. Physically consistent lie group mesh models for robot design and motion co-optimization. IEEE Robot Autom Lett. 2022;7(4):9501–8. doi:10.1109/LRA.2022.3189806.
18. Deng W, Zhang X, Zhou Y, Liu Y, Zhou X, Chen H, et al. An enhanced fast non-dominated solution sorting genetic algorithm for multi-objective problems. Inf Sci. 2022;585(5):441–53. doi:10.1016/j.ins.2021.11.052.
19. Giger M, Keller D, Ermanni P. AORCEA—an adaptive operator rate controlled evolutionary algorithm. Comput Struct. 2007;85(19–20):1547–61. doi:10.1016/j.compstruc.2006.12.002.

20. Xue Y, Zhu H, Liang J, Słowik A. Adaptive crossover operator based multi-objective binary genetic algorithm for feature selection in classification. Knowl Based Syst. 2021;227(3):107218. doi:10.1016/j.knosys.2021.107218.
21. Huang S, Colombo C, Bernelli-Zazzera F. Multi-criteria design of continuous global coverage walker and street-of-coverage constellations through property assessment. Acta Astronaut. 2021;188(1):151–70. doi:10.1016/j.actaastro.2021.07.002.
22. Xie S, Dong Y, Liang Z. Genetic programming method for satellite optimization design with quantification of multi-granularity model uncertainty. Aerosp Sci Technol. 2025;156(3):109764. doi:10.1016/j.ast.2024.109764.
23. Oltean M, Groşan C, Dioşan L, Mihăilă C. Genetic programming with linear representation: a survey. Int J Artif Intell Tools. 2009;18(2):197–238. doi:10.1142/s0218213009000111.
24. Wertz JR, Larson WJ. Space mission analysis and design. Microcosm Press. 2005 [cited 2026 Jun 5]. doi:10.1007/978-94-011-2692-2_1.
25. Hassan RA, Crossley WA. Multi-objective optimization of communication satellites with two-branch tournament genetic algorithm. J Spacecr Rockets. 2003;40(2):266–72. doi:10.2514/2.3942.
26. Li G, Zhang H, Tang G. Maneuver characteristics analysis for hypersonic glide vehicles. Aerosp Sci Technol. 2015;43(5):321–8. doi:10.1016/j.ast.2015.03.016.
27. Vince A. A framework for the greedy algorithm. Discrete Appl Math. 2002;121(1–3):247–60. doi:10.1016/S0166-218X(01)00362-6.
28. Zhai L, Feng S. A novel evacuation path planning method based on improved genetic algorithm. J Intell Fuzzy Syst. 2022;42(3):1813–23. doi:10.3233/jifs-211214.
29. Diouane Y, Gratton S, Vicente LN. Globally convergent evolution strategies. Math Program. 2015;152(1):467–90. doi:10.1007/s10107-014-0793-x.