

Multi-platform Cooperative Target Disposal Method Based on Dynamic Spatiotemporal Window and Multi-Agent Path Planning

Shilin Wei

Jiangsu Automation Research Institute, Liangyungang, China;

weishilin591@163.com

Abstract. Aiming at the problems of overlapping explosion hazard zones and large task waiting loss in the traditional batch mode of multi-platform delayed blasting cooperative operations in obstacle clearance and regional target disposal scenarios, this paper proposes a new path solving method integrating the idea of Multi-Agent Path Finding (MAPF). First, a spatiotemporal cylindrical model of explosion consisting of three stages: delayed detonation, hazard persistence, and safety recovery is established. The explosion hazard zone is modeled as a dynamic obstacle with a definite time window to characterize the time-varying characteristics of explosion hazards. On this basis, a multiple traveling salesman target assignment model with spatiotemporal window constraints is constructed, introducing coordination constraints such as safety of consecutive targets for the same platform, simultaneous disposal of multiple platforms, and a minimum waiting time insertion strategy. An improved Non-dominated Sorting Genetic Algorithm II (NSGA-II) is used to generate a Pareto optimal allocation scheme that balances task completion time and total voyage. For path planning, a hierarchical MAPF algorithm combining Conflict-Based Search (CBS) and spatiotemporal A* with hovering action is proposed to achieve the global optimal trade-off between detour and waiting costs. Simulation experiments show that the multi-platform cooperative algorithm based on dynamic spatiotemporal window and multi-agent path planning has higher reliability and safety when performing target disposal tasks.

Keywords: Multi-platform cooperation; Target disposal; Task assignment; Path planning.

1. Introduction

In various high-risk operation tasks such as obstacle clearance, dangerous target cleanup, and regional high-risk obstacle disposal, cluster cooperative operations of multiple types of unmanned platforms such as unmanned surface vehicles (USVs) and unmanned aerial vehicles (UAVs) are usually relied on. The platforms carry delayed blasting explosives to accurately destroy multiple scattered dangerous targets in the area. Each unmanned platform autonomously navigates to the designated target point, and after completing the accurate placement of explosives, it must immediately evacuate to a safe area quickly. After the preset fuze delay, the explosives trigger an explosion, which will form a large-area explosion hazard area with a range of hundreds of meters.

This paper decouples this problem into two steps: task assignment and path planning. The task assignment layer introduces coordination constraints such as safety of consecutive targets for the same platform, simultaneous disposal of multiple platforms, and a minimum waiting time insertion strategy to establish the corresponding relationship between task targets and platforms. On this basis, the path planning layer plans a safe path for the platform.

At present, the mainstream cluster task assignment algorithms at home and abroad can be divided into four categories, each with obvious shortcomings: First, traditional intelligent optimization algorithms^[1], centered on genetic algorithms, particle swarm optimization, etc., have mature principles and strong implementability, but poor dynamic real-time performance, which is difficult to adapt to the rapidly changing scenarios of explosion threats, easy to fall into local optimum, and insufficient convergence efficiency; although hybrid improved algorithms^[2] can make up for some defects, the computational overhead of dynamic re-planning is too large. Second, game theory cooperative assignment algorithms^[3] support distributed decision-making and have high adaptability to spatiotemporal constraints, but the modeling logic is complex, the

multi-objective weights are difficult to balance, and the iterative calculation cost for large-scale clusters is high. Third, deep reinforcement learning algorithms [4] have excellent dynamic adaptability and online decision-making speed, and can adapt to time-varying explosion scenarios, but they rely on massive training samples, have limited generalization, are prone to decision-making deviations under extreme working conditions, and have poor interpretability. Fourth, hierarchical cooperative optimization algorithms [5] reduce the solution complexity by decoupling hierarchical optimization, balancing accuracy and real-time performance, but lack pertinence, do not adapt to the exclusive spatiotemporal constraints of blasting operations, and the modeling and iterative design are difficult.

Multi-platform path planning can essentially be reduced to the Multi-Agent Path Finding (MAPF) problem [6][7]. The mainstream solution framework of classic MAPF is Conflict-Based Search (CBS) [8]. However, most existing MAPF researches carry out discrete motion planning based on regular grid environments [9], assuming that the target position is static and unchanged, the environment state is not affected by the agent's behavior, and the obstacle elements are pre-known fixed or external time-varying objects.

Different from traditional scenarios, in the process of multi-platform cooperative blasting operation, the platform's task behavior of placing explosives will actively generate time-varying obstacles with time window characteristics in the environment, that is, the explosion no-fly zone. The start and end times of such obstacles are directly related to the time when the corresponding platform arrives at the target point, thus forming complex self-coupling and mutual coupling constraints: the track timing of a single platform will determine the effective interval of its own derived obstacles, forming a self-coupling relationship; the planning results of different platforms restrict each other through their respective generated explosion windows, forming a mutual coupling relationship. The above characteristics completely break the environmental assumptions of classic MAPF, which greatly increases the difficulty of problem solving. At present, few studies combine the agent's self-generated time window obstacle with the MAPF framework.

In summary, for the special scenario of unmanned platform cluster cooperative blasting operation, this research breaks through the inherent limitations of traditional static obstacle avoidance modeling, and constructs a complete cooperative optimization system adapted to the spatiotemporal dynamic constraints of explosions from the two core dimensions of task assignment and path planning. By introducing spatiotemporal window constraints and minimum waiting strategy, it effectively solves the pain points of blind detour and resource waste of traditional algorithms; at the same time, it improves the MAPF solution framework, and for the first time realizes the scenario adaptation to the agent's self-generated dynamic obstacles, filling the research gap in this special scenario. This research not only provides theoretical support and implementable technical solutions for the safe and efficient development of cluster blasting operations, but also provides new research ideas and solutions for the multi-agent cooperative decision-making problem in scenarios with strong spatiotemporal coupling and self-coupling dynamic obstacles, laying a technical foundation for the large-scale application of unmanned cluster technology in special operation scenarios.

2. Problem Description and Mathematical Model

The mission scenario of this paper focuses on multi-unmanned platform cooperative blasting operations. Given a two-dimensional bounded mission area $W \subset \mathbb{R}^2$, there are N_T static targets randomly distributed in the area, whose positions have been accurately located through prior reconnaissance and remain unchanged throughout the mission execution. The position coordinate of target i is $\mathbf{p}_i = (x_i, y_i)$. To efficiently dispose of batch targets, K homogeneous unmanned platforms are deployed. The maximum speed of all platforms is v_{max} , and they have precise hovering capability. Before the mission starts, all platforms are pre-deployed at their respective initial starting positions $\mathbf{s}_k$. After completing the mission, each platform needs to return to its initial

standby position for subsequent platform recovery, ammunition replenishment, and the next round of mission deployment.

2.1 Spatiotemporal Window Model of Explosion Danger

Each platform can carry multiple delayed explosives and has the capability to continuously dispose of multiple targets. After the platform arrives at target i , it places the explosive and immediately leaves the position at maximum speed. The explosive placement time is recorded as t_i , which is determined by the actual time the platform arrives at that target point. The fuse time is fixed as T_{fuse} . The explosion forms a circular danger zone centered at the placement point with a radius of R , and the danger duration is T_{danger} . That is, within the time interval $[t_i + T_{fuse}, t_i + T_{fuse} + T_{danger}]$, this circular area is an absolute no-entry zone; no platform may enter its interior or boundary. Before this interval (within T_{fuse} after placement) and after (danger dissipation), the area can be safely traversed.

The explosion danger generated by any target i can be defined in the spatiotemporal domain as a cylindrical obstacle. The projection of this obstacle in two-dimensional space is a circle centered at $\mathbf{p}_i$ with radius R , and its span on the time axis is the danger window. Then, the danger spatiotemporal region of target i is:

$$\mathcal{O}_i = (x, y, t) \in W \times \mathbb{R}^+ | \sqrt{[(x - x_i)^2 + (y - y_i)^2]} \leq R, t \in [t_i + T_{fuse}, t_i + T_{fuse} + T_{danger}] \quad (1)$$

Since explosion effects are superimposable and independent of each other, the dynamically changing danger zone during the entire mission is the union of the cylinders generated by all targets:

$$\mathcal{O}(t) = \bigcup_{i \in T} \mathcal{O}_i(t) \quad (2)$$

where $\mathcal{O}_i(t)$ is the spatial cross-section of $\mathcal{O}_i$ at time t .

2.2 Mathematical Formalization of the Cooperative Mission Planning Problem.

This paper decouples the original strongly coupled big problem into two sub-problems that can be solved independently yet interact cooperatively, reducing computational complexity while ensuring constraint integrity and solution feasibility.

2.2.1 Target Allocation and Temporal Coordination

Target allocation is the top-level decision of the cooperative mission. Its core is to determine the task division of each platform, i.e., to determine an ordered target visit list $\pi_k = (s_{k,1}, s_{k,2}, \dots, s_{k,m_k})$ for each platform k , where $s_{k,j} \in T$ is the target number of the j -th visit of platform k , and m_k is the number of targets allocated to platform k . The allocation process must satisfy completeness and exclusivity: $\bigcup_{k=1}^K \pi_k = T$, and the target sets of any different platforms are disjoint.

Each visited target will have a corresponding placement time $t_{k,j}$. This time not only depends on the platform's navigation distance and movement speed, but also needs to satisfy the following two key types of temporal coordination constraints:

(1) Same-platform safety waiting constraint:

The placement times for two consecutive targets on the same platform must satisfy:

$$t_{k,j+1} - t_{k,j} \geq T_{danger} \quad (3)$$

(2) Cross-platform cooperative time constraint: For target $s_{k,j}$ of platform k and target $s_{k',j'}$ of platform k' , if $\|p_{s_{k,j}} - p_{s_{k',j'}}\| < R$, then it must satisfy:

$$t_{k,j} - t_{k',j'} \geq T_{danger} \quad (4)$$

This ensures that the placement actions of different platforms do not interfere with each other, avoiding cross-platform safety risks.

2.2.2 Path Planning with Dynamic Constraints

After determining the target allocation plan and the corresponding placement times, it is necessary to generate a continuous, safe spatiotemporal trajectory $p_k(t): [0, C_k] \rightarrow W$ for each platform, C_k is the mission completion time of platform k . This trajectory must satisfy the following strict constraints:

(1) Boundary conditions: The start and end points of the trajectory strictly match the platform's initial position and the recovery position after mission completion.

(2) Target visit condition: The platform must visit each target point strictly according to the allocated target sequence and accurately arrive at the target location at the designated placement time.

(3) Kinematic constraints: The platform's movement cannot exceed its own maximum maneuverability. For any two moments $t_2 > t_1$, $\|p_k(t_2) - p_k(t_1)\| \leq v_{max} \cdot |t_2 - t_1|$.

(4) Explosion safety constraint: The spatiotemporal trajectory $p_k(t) = (x_k(t), y_k(t))$ of any platform k must satisfy the following safety constraint:

For any $t \geq 0$ and any target i , if $t \in [t_i + T_{fuse}, t_i + T_{fuse} + T_{danger}]$, then $\sqrt{[(x_k(t) - x_i)^2 + (y_k(t) - y_i)^2]} > R$.

(5) Platform anti-collision constraint: For any two different platforms $k \neq l$, when $t \geq 0$, $\sqrt{[(x_k(t) - x_l(t))^2 + (y_k(t) - y_l(t))^2]} \geq d_{safe}$, where d_{safe} is the minimum safe distance between platforms.

2.2.3 Formalization of the Bi-Objective Optimization Problem

Multi-platform cooperative disposal missions usually pursue two conflicting performance indicators: one is time efficiency, measured by the maximum mission completion time C_{max} , reflecting the speed of the entire fleet completing all tasks; the other is energy efficiency, measured by the total voyage distance of all platforms L_{total} , directly affecting endurance and fuel consumption. Specifically defined as:

$$C_{max} = \max_k C_k, \quad L_{total} = \sum_k L_k \quad (5)$$

3. Multi-Objective Allocation Based on NSGA-II

This paper adopts the Non-dominated Sorting Genetic Algorithm with elite strategy (NSGA-II) for solving.

3.1 Chromosome Encoding and Decoding

This paper adopts a permutation encoding method. A chromosome is a permutation of all target numbers, denoted as $\pi = (g_1, \dots, g_{N_T})$, where $g_j \in T$ and all distinct, with length N_T . The decoding process converts the chromosome into target sequences and estimated placement times for each platform. A shortest-path greedy cut decoding strategy is designed, whose core idea is: process targets one by one according to the permutation order, each time assigning the current target to the platform that "can complete disposal the earliest". When calculating the arrival time, if the distance between the platform's previous target and the current target is less than R , a mandatory wait of $T_{fuse} + T_{danger}$ is automatically inserted.

3.2 Constraint Handling and Fitness Calculation

After decoding the preliminary allocation plan, constraint handling and time simulation are required to verify the actual feasibility under explosion safety constraints and calculate the real placement times and mission costs.

3.2.1 Time simulation and waiting insertion

For each platform's sequence, simulate execution from the start point in order. For two adjacent targets $i \rightarrow j$ in the sequence, handle according to the same-platform target safety distance constraint: If $d_{ij} \geq R$, no additional waiting needed, the platform can travel directly from i to j . Else, j is within the explosion radius of i , the platform can only enter j after the explosion of i ends. The required waiting time is at least $\Delta T_{sync} = T_{fuse} + T_{danger} - T_{ij}^L$, where T_{ij}^L is the time already consumed by the platform evacuating from i to the safety boundary and traveling to j . This waiting period is explicitly inserted during time simulation.

3.2.2 Multi-platform cooperative constraint check

Check whether there exist pairs of targets disposed simultaneously by different platforms that are too close $d_{ij}^V < 2R$ and placement time difference $< \Delta T_{sync}$. If violated, the later placement time is forcibly adjusted to stagger at least ΔT_{sync} from the earlier one, and this adjustment amount is counted into the constraint violation degree.

3.2.3 Fitness output

After the above processing, for each platform k , the actual earliest executable placement time sequence, task completion time C_k , and voyage distance L_k are obtained. The fitness of an individual consists of a two-dimensional objective vector (f_1, f_2) and constraint violation degree ϕ :

$$\begin{cases} f_1 = C_{max} = \max_k C_k & (\text{maximum mission completion time}) \\ f_2 = L_{total} = \sum_k L_k & (\text{total voyage distance}) \\ \phi = \sum \text{all forced time adjustments} \end{cases} \quad (6)$$

3.2.4 Constraint domination rule

In the non-dominated sorting and multi-objective comparison of NSGA-II, the constraint domination rule is adopted to handle infeasible solutions. For two individuals x and y :

- If x is feasible and y is infeasible $\phi_y > 0$, then x dominates y ;
- If both are infeasible and $\phi_x < \phi_y$, then x dominates y ;
- If both are feasible, conventional Pareto dominance is compared.

This rule drives the population to quickly evolve from the infeasible region to the feasible region, obtaining feasible solutions early in the evolution and continuously optimizing the two objectives in subsequent generations.

4. Multi-Platform Cooperative Path Planning Based on MAPF

Multi-platform spatiotemporal cooperative path planning is essentially a multi-agent path finding problem with dynamic obstacle constraints. This paper treats each unmanned platform as an independent agent $a_k (k = 1, 2, \dots, K)$, whose path is defined as a set of spatiotemporal point sequences increasing in time, $p_k = [(x_0, y_0, t_0), (x_1, y_1, t_1), \dots, (x_{M_k}, y_{M_k}, t_{M_k})]$, satisfying constraints such as boundary conditions, target visits, motion constraints, platform collision avoidance, and explosion danger avoidance. Among them, the explosion window is determined by the placement time, and the path choice affects the feasible space of itself and other platforms, forming self-coupling and inter-coupling.

Conflict-Based Search (CBS) is a classic algorithm for solving MAPF problems^[10]. Its core idea is to decompose multi-agent planning into two levels: the high level is responsible for detecting conflicts between paths and imposing constraints^[11], and the low level independently plans paths for each agent under constraints^[12]. CBS can guarantee finding the global optimal solution (under a given objective function) in discrete states, and is more computationally efficient than centralized joint planning^[13].

4.1 Low-Level Planning

The low-level planner plans a safe spatiotemporal path from the start point through the target sequence to the end point for a single agent under a given constraint set Ω_k ^[14]. The standard A* algorithm only searches in two-dimensional space and cannot handle time-varying obstacles and hovering waiting. This paper designs a spatiotemporal A* algorithm with hovering actions, expanding the search space from a 2D plane to 3D spatiotemporal.

Assume the search state $s = (x, y, t)$, where (x, y) are discrete coordinates after gridding (resolution r), and t is a discrete time step (step size Δt). To ensure neighborhood movement reachability, the grid resolution and time step must satisfy $r \leq v_{max} \cdot \Delta t$. The cumulative cost from the start point to the current state s is defined as: $g(s) = w_t t + w_d d_{trav}$.

Here, d_{trav} is the cumulative voyage distance, and w_t and w_d are weight coefficients for time and distance, respectively. The heuristic function is $h(s) = w_t \cdot (\|p_{goal} - p_{current}\| / v_{max})$. During expansion, states violating constraints or falling into $\mathcal{O}(t)$ are discarded. This algorithm makes hovering an explicit decision. When a target is covered by danger, direct entry is pruned; hovering advances time until the danger ends, and entry to the target becomes legal again. A* automatically compares the "waiting cost" and "detour cost" to select the optimal strategy without preset rules.

4.2 High-Level Planning

The node cost of the high-level search is a two-dimensional vector (C_{max}, L_{total}) , constituting a multi-objective optimization problem. Its core is to maintain a constraint tree, where each node in the tree represents a set of constraints and the path set of all agents satisfying these constraints^[15]. The high-level search adopts best-first search, expanding nodes in ascending order of their global SOC cost, thus ensuring the final found solution is optimal.

4.3 Conflict Detection and Resolution

Three types of conflicts are defined: platform collision (spatial distance less than d_{safe}), imposing vertex/edge constraints; platform-explosion conflict (path entering the explosion cylinder), imposing spatiotemporal forbidden entry constraints or delayed placement constraints; target point self-conflict (adjacent target distance less than R) and arriving too early), imposing a lower bound constraint on arrival time. The earliest occurring conflict is resolved first.

5. Simulation Experiments and Analysis

To comprehensively verify the effectiveness and superiority of the proposed method based on dynamic spatiotemporal windows and multi-agent path finding for target disposal, this section sets up multiple sets of simulation experiments to systematically evaluate the algorithm.

5.1 Experimental Setup

Set $2000 \text{ m} \times 2000 \text{ m}$ sea area, randomly generate $N_T = 15$ targets and 3 platforms, coordinates follow a uniform distribution. Explosion radius $R=300 \text{ m}$, fuse time $T_{fuse} = 15 \text{ s}$, danger duration $T_{danger} = 20 \text{ s}$. Maximum speed is $v_{max} = 30 \text{ m/s}$, inter-platform safe distance $d_{safe} = 10 \text{ m}$.

NSGA-II population size $N=50$, evolution 100 generations, crossover probability 0.9, mutation probability 0.1. Spatiotemporal A* grid resolution 5 m, time step $\Delta t = 0.5 \text{ s}$, weights $\omega_t = 1$, $\omega_d = 0.001$. CBS high-level search upper limit 500 nodes.

This paper adopts a "hierarchical decoupling—cooperative optimization" solving paradigm: first, at the upper level, a multi-objective evolutionary algorithm solves the target allocation problem considering explosion radius constraints, comprehensively considering (mission completion time, path distance) to obtain a set of Pareto optimal task division schemes (see Fig. 1); then, at the lower level, the path planning of each scheme is transformed into a multi-agent path finding problem with

dynamic obstacles, solved jointly using Conflict-Based Search (CBS) and spatiotemporal A* (see Fig. 2 for task execution results). This technical route, while ensuring completeness and safety, achieves effective solutions for high-density target scenarios, which traditional static obstacle methods cannot achieve.

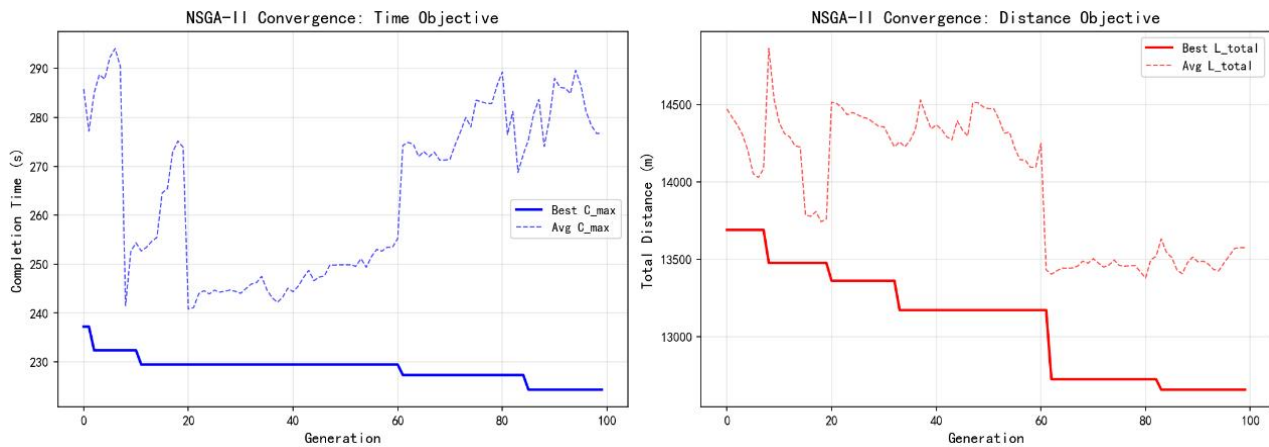
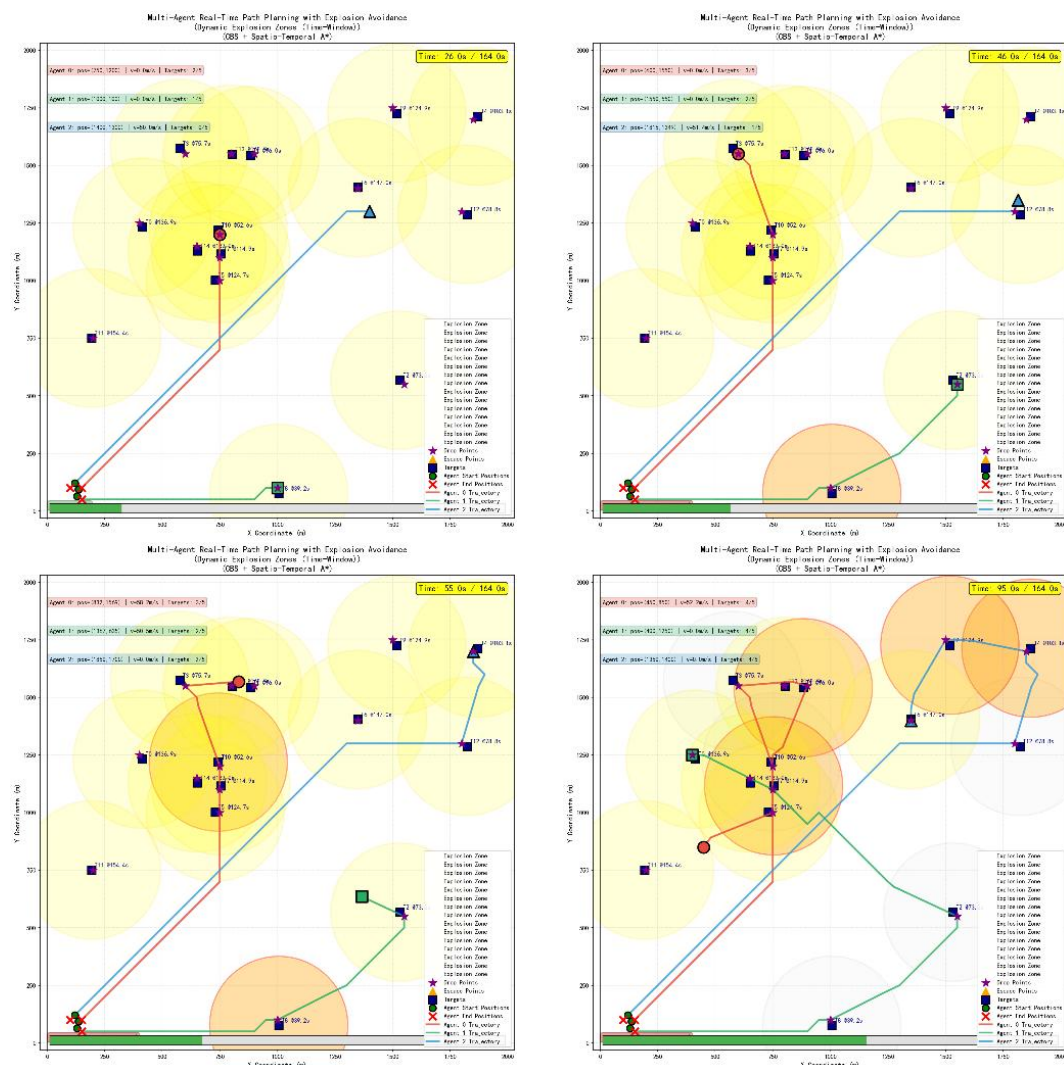


Fig. 1 Multi-objective allocation iteration process based on NSGA-II



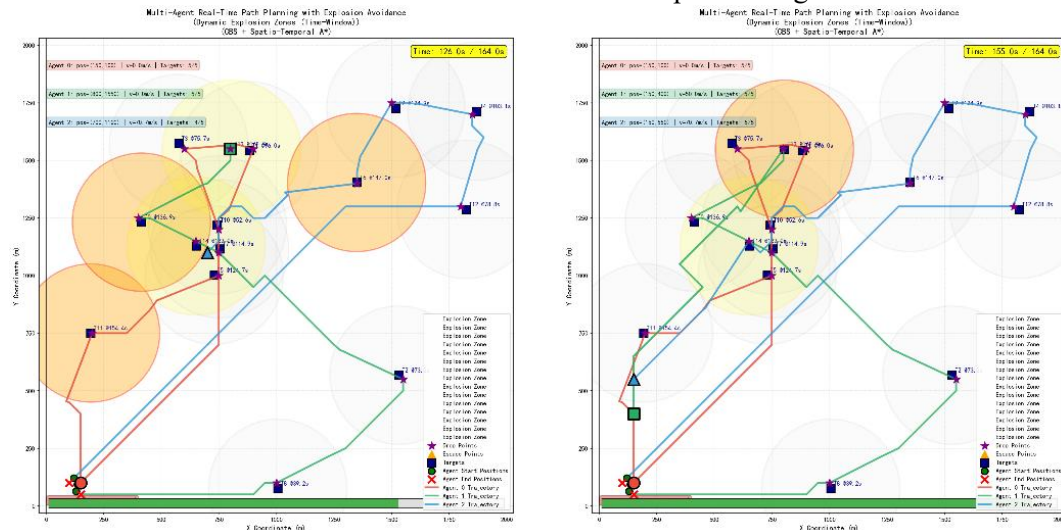


Fig. 2 Execution phase diagram of MAPF-CBS algorithm

5.2 Comparative Methods

To verify the effectiveness of the proposed method, four comparative approaches are established: conventional static obstacle method(Static), dynamic-window no-wait method (NW-Dyn), priority-sequenced planning method(Priority-Based), and the proposed method(MAPF-CBS).To conveniently compare the operation of the four algorithms under different task scales, four scenarios of different scales are set up: small scale (900×900m, 4 targets, 2 platforms), medium scale (1200×1200m, 8 targets, 3 platforms), large scale (1500×1500m, 15 targets, 5 platforms), and extra-large scale (2000×2000m, 25 targets, 8 platforms). Platform positions and target positions are randomly generated, coordinates follow uniform distribution, and each scenario is run 100 times. Task feasibility rate, average task completion time, and average planning time are statistically recorded.

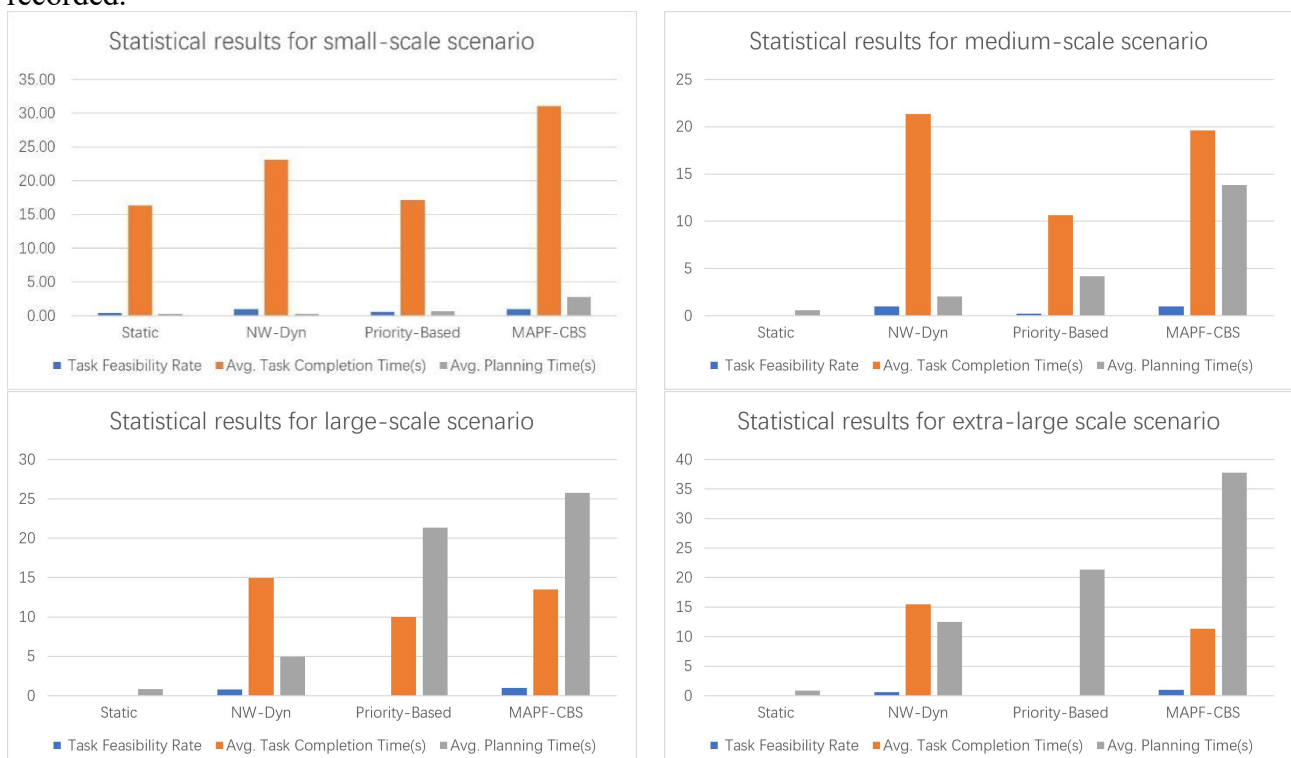


Fig.3 Statistical results

Synthesizing the experimental results of the four scale scenarios, from small and medium to extra-large scale, the MAPF-CBS method proposed in this paper is the only algorithm that always maintains a 100% task feasibility rate, holding an absolute advantage in extra-large scenarios where both Static and Priority-Based completely fail. Although MAPF-CBS has a slight efficiency loss due to preventive waiting in small-scale scenarios, as the scenario scale and density increase, its task completion time advantage gradually appears and expands, proving the necessity and high efficiency of global coordination and hovering waiting strategies in high-density scenarios. The planning time of MAPF-CBS increases with scale, but remains within a reasonable range of tens of seconds in extra-large scale scenarios, possessing practical value as an offline planner.

6. Summary

This paper proposes a novel planning framework integrating a dynamic spatiotemporal window explosion model with MAPF for the multi-target cooperative disposal problem of multi-platform delayed explosive placement. The explosion danger zone is modeled as a time-limited spatiotemporal cylindrical obstacle, breaking the traditional static obstacle assumption and fundamentally solving the no-solution problem of path planning under large-radius explosions. Explosion radius constraints and a minimum waiting strategy are introduced in target allocation, and NSGA-II is used for multi-objective optimization, providing decision-makers with a flexible time-voyage trade-off scheme. The multi-platform path planning is transformed into a MAPF problem, and a CBS-spatiotemporal A* hierarchical algorithm is designed. The low-level planning explicitly includes hovering actions, automatically achieving optimal trade-off between detour and waiting; the high level achieves global coordination through conflict resolution. The superiority and contribution of each module have been verified through extensive simulation experiments.

References

- [1] Wang Chuanyun, Xing Jiaqing, Wang Tian, et al. Multi-strategy fusion seagull optimization algorithm for multi-UAV cooperative task allocation [J/OL]. *Acta Automatica Sinica*, 1-15 [2026-06-01]. <https://doi.org/10.16383/j.aas.c250754>.
- [2] Bai Xiaoshan, Zhang Miaosen, She Anqi, et al. Task allocation method for multiple heterogeneous unmanned surface vehicles based on improved genetic algorithm [J/OL]. *Information and Control*, 1-14 [2026-06-01]. <https://doi.org/10.13976/j.cnki.xk.2025.3773>.
- [3] Zhang Chun, Wu Yongjiang, Wang Xuemei. A game task allocation algorithm for heterogeneous UAV swarms [J]. *Journal of Chinese Mini-Micro Computer Systems*, 2026, 47(4): 793-801. DOI:10.20009/j.cnki.21-1106/TP.2025-0068.
- [4] Wei Haoming, Wang Zhengjie, Wang Yue, et al. Cooperative target allocation method against heterogeneous UAV swarms based on reinforcement learning [J]. *Transactions of Beijing Institute of Technology*, 2026, 46(5): 527-533. DOI:10.15918/j.tbti1001-0645.2026.003.
- [5] Zhao Wenzhi, Hu Tao, Wu Di, et al. Dynamic task allocation method for multiple UAVs based on collaborative matching attention [J/OL]. *Acta Armamentarii*, 1-12 [2026-06-01]. <https://link.cnki.net/urlid/11.2176.TJ.20260311.1720.006>.
- [6] Zhang Jiapeng, et al. A survey of multi-agent path finding techniques [J]. *Journal of Beijing University of Technology*, 2023.
- [7] Li Xiaoming, et al. A multi-robot path planning algorithm based on improved conflict-based search [J]. *Control and Decision*, 2023.
- [8] Sharon G, Stern R, Felner A, et al. Conflict-Based Search For Optimal Multi-Agent Path Finding [C]//AAAI. 2015.
- [9] Boyarski E, Felner A, Stern R, et al. ICBS: The Improved Conflict-Based Search Algorithm for Multi-Agent Pathfinding [C]//Symposium on Combinatorial Search. 2015.
- [10] Barer M, Sharon G, Stern R, et al. Suboptimal Variants of the Conflict-Based Search Algorithm for the Multi-Agent Pathfinding Problem [C]//SoCS. 2014.

- [11] Li J, Ruml W, Koenig S. EECBS: A Bounded-Suboptimal Search for Multi-Agent Path Finding [C]//AAAI. 2021.
- [12] Li J, Harabor D, Stuckey P J, et al. Pairwise Symmetry Reasoning for Multi-Agent Path Finding Search [J]. Artificial Intelligence, 2021.
- [13] Maliah A, Atzmon D, Felner A. Minimizing Makespan with Conflict-Based Search for Optimal Multi-Agent Path Finding [C]//AAMAS. 2025.
- [14] Wang Q, Veerapaneni R, Wu Y, et al. MAPF in 3D Warehouses: Dataset and Analysis [C]//ICAPS. 2024.
- [15] Stern R. Multi-Agent Path Finding – an Overview [M]//Lecture Notes in Computer Science. 2019.