



Research article

A fuzzy auction genetic algorithm for the vehicle-UAV collaborative delivery problem

Chunping Jiang, Dingjie Wu* and Chang Cai

School of Air Traffic Control, Civil Aviation Flight University of China, Chengdu, 618307, CHINA

* **Correspondence:** Email: hellowoody@cafuc.edu.cn; Tel: 13778220465.

Abstract: Efficient task allocation in vehicle-UAV collaborative delivery is challenged by heterogeneous resource capabilities and traffic uncertainty. Auction-based schemes often conflate feasibility screening with execution-level selection and rarely connect fuzzy bidding to subsequent route refinement. To address this issue, we proposed an Improved Fuzzy Auction-Genetic Algorithm (IFAGA) for congestion-aware task allocation and routing optimization in urban logistics. IFAGA combined a two-round fuzzy auction with collaborative evolutionary optimization. The first round screened depots using location proximity, capability matching, and time-window feasibility. The second round selected feasible vehicles or UAVs within the winning depot. The auction output then initialized a genetic stage that repaired assignments and improved workload balance and route efficiency under traffic disturbance. On a fixed 15-task benchmark with 20 independent runs, comparisons against ALNS, Classical-DP, NSGA-II, VNS, and a discrete particle swarm optimization baseline (DPSO) showed a trade-off among makespan, total cost, obstacle avoidance, and runtime. A separate event-based rolling horizon experiment with staged task release evaluated completion, lateness, and re-optimization overhead. Across additional random scalability instances, IFAGA achieved the lowest average makespan among the tested task scales, while cost and obstacle avoidance varied by baseline and scale.

Keywords: vehicle-UAV collaboration; auction mechanism; genetic algorithm; multi-objective path optimization

Mathematics Subject Classification: 90B06, 90C59, 68W50, 68T37, 91B26

1. Introduction

The integration of smart-city infrastructure and e-commerce has made warehousing and transportation systems more complex and more dependent on timely, resilient operations. Vehicle routing remains a core problem in logistics because it directly affects transport efficiency and operating cost. In its classical form, a fleet serves geographically distributed customers and returns to a central depot. Later variants, such as the vehicle routing problem with time windows and the multi-depot vehicle routing problem, extend this setting to multiple depots and tighter operational constraints. For example, Arenas-Vasco et al. [1] studied a periodic capacitated vehicle routing problem with multiple depots, a heterogeneous fleet, and hard time windows. Cao et al. [2] considered order fulfillment planning with crowdsourcing integration. Compared with single-resource distribution systems, heterogeneous collaborative systems are more flexible and efficient [3]. However, they also make task allocation and path planning harder under uncertain traffic, time-window constraints, and heterogeneous capabilities [4]. Auction-based mechanisms have attracted attention because they support decentralized task allocation and local decision-making. Abusalama et al. [5] addressed the winner determination problem in reverse combinatorial auctions. De Ryck et al. [6] developed decentralized task allocation for industrial AGV systems with routing constraints. Liu et al. [7] proposed an auction-based coordination strategy for task-constrained multi-agent stochastic planning with submodular rewards. Quinton et al. [8] studied communication-preserving bids in market-based task allocation.

However, most schemes rely on single-round bidding with simplified metrics, so they do not separate feasibility screening from execution efficiency. Ozturk et al. [9] considered adaptive market-based dynamic task allocation under environmental uncertainty. Wang et al. [10] proposed an adaptive distributed auction algorithm for multi-AUV task assignment. Alshaboti and Baroudi [11] developed fuzzy auction-based and adaptive multi-threshold approaches for multi-robot task allocation. Tao and Li [12] improved an auction algorithm for task allocation in RMFS. Despite these advances, heterogeneous systems can still face suboptimal assignments, overload, and limited robustness under environmental uncertainty.

This study makes three related contributions. First, we separate depot-level feasibility screening from execution-level resource selection in heterogeneous vehicle-UAV delivery. Second, we integrate membership-based weighted scoring, auction initialization, assignment repair, and route refinement in one optimization workflow. Third, for the experiments, we evaluate this integration under fixed static benchmarks, randomly generated scalability instances, and staged order-release controls. The results indicate a competitive trade-off among makespan, routing cost, obstacle avoidance, feasibility, and runtime.

The remainder of this study is organized as follows: In Section 2, we review related work on heterogeneous task allocation, fuzzy decision-making, and GA-based path optimization. In Section 3, we formulate the problem and mathematical model. In Section 4, we present the algorithmic framework. In Section 5, we report the experiments. In Section 6, we conclude the study and outline future work.

2. Literature review

Prior work on heterogeneous task allocation mainly follows two lines: centralized optimization and decentralized market-based allocation. Centralized methods can produce strong solutions on small instances, but scalability and robustness become limiting factors in dynamic settings [13,14].

Auction-based mechanisms improve autonomy and scalability. Li et al. [15] combined distributed auction with IBPSO for scalable and energy-efficient task allocation in Industry 4.0. Choi et al. [16] proposed consensus-based decentralized auctions for robust task allocation. Wang et al. [17] applied combinatorial auction to multi-agent single-machine scheduling. Wang et al. [18] developed a hierarchical auction-based mechanism for real-time resource allocation in cloud robotic systems. Ren et al. [19] formulated a bidding game model for power generation companies based on first-price sealed auction. However, many of these schemes still use single-round bidding and a flat decision structure, which makes it difficult to distinguish feasibility screening from execution selection.

Fuzzy logic has been used to represent uncertainty in task allocation and scheduling. Garibaldi and Ozen [20] studied uncertain fuzzy reasoning in modeling expert decision making. Wu et al. [21] reviewed interval type-2 fuzzy sets and systems. Sekhi et al. [22] optimized fuzzy logic systems to enhance decision-making in uncertain domains. It is well suited to soft constraints and ambiguous decisions, but many studies stop at the evaluation stage and do not carry fuzzy information into subsequent optimization. As a result, the advantages of fuzzy modeling are only partially realized.

Genetic algorithms have also been widely applied to routing and scheduling problems because they offer flexible global search and can accommodate practical constraints such as time windows, multi-depot operations, and heterogeneous fleets. Jiang et al. [23] proposed a robustness division based multi-population evolutionary algorithm for vehicle routing problems with uncertain demand. Rezaei et al. [24] combined genetic local search with a multi-population imperialist competitive algorithm for the capacitated vehicle routing problem. Zhu [25] developed a diversity-controlling adaptive genetic algorithm for the vehicle routing problem with time windows. Related studies on vehicle-UAV collaboration often focus on cooperation strategies, energy models, or operational constraints, but they still tend to separate task allocation from route optimization. Hybrid and distributed baselines also provide useful comparisons, including NSGA-II, GA-SA, ALNS [26–28], real-time multi-UAV task assignment [29], consensus-based auction (CBBA), and MARL [16,30]. These methods show useful strengths, yet each leaves a different trade-off between solution quality, computational cost, and robustness.

Taken together, the literature points to two unresolved issues. First, hierarchical task allocation is often missing, so capability screening and execution selection remain conflated. Second, fuzzy uncertainty modeling is seldom integrated with path optimization in a single workflow. These gaps motivate the double-layer fuzzy auction and genetic optimization framework proposed in this study. This leads to the central question examined here: Can a two-round fuzzy auction coupled with post-auction genetic repair improve assignment-route coordination while preserving feasibility under congestion disturbance.

3. Mathematical formulation of the resource-task allocation problem

The allocation process follows three stages. First, tasks enter a two-round fuzzy auction that screens depots by proximity, capability, and time-window feasibility. Second, the winning depot performs execution-level competition among feasible vehicles and UAVs. Third, the auction output is repaired and refined by the genetic stage to improve route quality and workload balance under traffic disturbance. For reproducibility, the notation used in this section is standardized as follows. The task set is denoted by $T = \{1, \dots, N\}$, where $N = |T|$ is the number of delivery tasks. The depot, vehicle, and UAV sets are denoted by W , V , and U , respectively. For task i , the input attributes include location,

demand, time window, and base price P_i^{base} . The road-blockage penalty factor is denoted by ϕ_{block} , and the post-repair feasibility score by $F_{\text{feas}}(\mathbf{S})$.

Table 1. Fuzzy indicator conversion used in the membership-based bidding procedure.

Indicator group	Input basis	Normalization direction	Score range
Capability matching	Resource-to-demand ratio	Values closer to or above the task requirement receive higher membership through Eq (1)	[0, 1]
Distance, cost, risk, and traffic	Distance, operating cost, airspace risk, or traffic factor	Lower values receive higher suitability scores after bounded normalization	[0, 1]
Time and urgency	Time-window slack, urgency, and arrival feasibility	Feasible and more urgent assignments receive higher scores within the feasible interval	[0, 1]
Capacity, endurance, and range	Residual capacity, UAV endurance, or sortie distance	Feasible margins receive higher scores; infeasible assignments are excluded or penalized	[0, 1]
Task matching and depot context	Task-resource compatibility and operating context	Higher compatibility and more suitable depot context receive higher scores	[0, 1]

3.1. Asymmetric gaussian function

The auction uses a piecewise membership function; for each indicator, the procedure first produces a value in the unit interval. Capability indicators are expressed as resource-to-demand ratios and mapped with the asymmetric Gaussian function in Eq (1). Other indicators are converted directly to bounded scores by their corresponding distance, capacity, cost, time-window, traffic, endurance, risk, urgency, task-matching, or depot-context functions. Table 1 Fuzzy indicator conversion, which reports the input basis, normalization direction, and score range for the indicator groups. The weighted sum of these membership values gives the candidate fitness, and the bid is then obtained by multiplying the fitness by the task base price. No fuzzy rule base or defuzzification step is used. Here, x is the normalized capability-to-demand ratio. We use $\sigma_1 = 0.2$ for insufficient capability and $\sigma_2 = 1$, for capability at or above the requirement.

$$g(x) = \begin{cases} \exp\left(-\frac{(x-1)^2}{2\sigma_1^2}\right), & x < 1, \\ \exp\left(-\frac{(x-1)^2}{2\sigma_2^2}\right), & x \geq 1. \end{cases} \quad (1)$$

3.2. Distance representation and feasibility constraints

The distance model is separated by resource type. The great-circle distance in Eq (2) is used for UAV air-distance estimation. In this equation, $\Delta\phi$ and $\Delta\lambda$ are latitude and longitude differences in

radians, and $R = 6371$ km is the Earth radius. Geographic coordinates are converted from degrees to radians before evaluating the trigonometric functions. Vehicle routing instead uses a ground-distance model with road detour and obstacle factors. Thus, vehicle routing and UAV range checks do not use the same distance representation.

$$d_{haversine}(u,v)=2R \cdot \arcsin \left(\sqrt{\sin^2 \left(\frac{\Delta \varphi}{2} \right) + \cos(\varphi_u) \cdot \cos(\varphi_v) \cdot \sin^2 \left(\frac{\Delta \lambda}{2} \right)} \right) \quad (2)$$

$$d_g(p_i, p_j) = d_{haversine}(p_i, p_j) \times \delta_{road} \times (1 + \emptyset_{block}(p_i, p_j)) \quad (3)$$

The hard constraints are written as follows: Eq (4) limits each task to at most one execution resource; complete final solutions are required to achieve an assignment rate of 1. Eq (5) constrains the cumulative task weight assigned to each ground vehicle over the planning horizon. Eq (6) imposes UAV payload feasibility at the sortie level, so every task assigned to a UAV must individually satisfy the UAV payload limit. Eq (7) checks each UAV task as an independent depot-task-depot flight. Thus, a UAV may serve multiple tasks sequentially, but it does not carry multiple tasks on one flight. Binary variables x_{ij} and y_{ik} indicate whether task i is assigned to resource j and depot k , respectively, as defined in Eq (8). These constraints are enforced in auction screening and in the repair stage.

$$\sum_{j \in \text{VUU}} x_{ij} \leq 1, \forall i \in T \quad (4)$$

$$\sum_{i \in T_j} w_i \leq C_j, \forall j \in \text{VUU} \quad (5)$$

$$w_i \leq C_j, \forall i \in T_j, \forall j \in U \quad (6)$$

$$d_{ij} + d_{ji} \leq R_j, \forall i \in T, \forall j \in U \quad (7)$$

$$x_{ij}, y_{ik} \in \{0, 1\} \quad (8)$$

3.3. Depot-level bidding formulation

In the first auction layer, each task broadcasts its identifier, location, demand profile, time window, and base price to all depots. Each depot k computes a depot-level fuzzy fitness Eq (9), where the three indicators are location proximity, capability matching, and time-window feasibility. A bid is generated only when the fitness exceeds a preset threshold Eq (10). The winning depot is selected by the highest valid bid Eq (11).

$$f_{ik}^D = \sum_{m=1}^3 w_m^D \mu_m^D(i, k) \quad (9)$$

$$B_{ik}^{(I)} = f_{ik}^D P_i^{\text{base}} I[f_{ik}^D > \theta_I] \quad (10)$$

$$k^* = \arg \max_k B_{ik}^{(I)} \quad (11)$$

3.4. Resource-level allocation formulation

After the first round, the depot with the largest valid bid $B_{ik}^{(1)}$ becomes the candidate service depot for task i . The second round then evaluates resources belonging to that depot. For vehicles, Eq (12), the five indicators are path synergy, residual-capacity adequacy, economic-cost suitability, time-window slack, and traffic index. For UAVs, Eq (13), the five indicators are endurance adequacy, airspace-risk suitability, time urgency, task matching, and depot-context dependency. The last indicator represents a land-use and operating-context score in the model; it does not represent dependence on a carrier vehicle or mother ship. No vehicle-UAV launch, landing, or synchronization relationship is assumed in this model. The weighted membership scores define f_{ij}^V and f_{ij}^U , and Eq (14) converts them into second-round bids.

The second-round auction provides only an initial assignment; the final executed solution is obtained after repair and route optimization.

$$f_{ij}^V = \sum_{m=1}^5 w_m^V \mu_m^V(i,j) \quad (12)$$

$$f_{ij}^U = \sum_{m=1}^5 w_m^U \mu_m^U(i,j) \quad (13)$$

$$B_{ij}^{(2)} = 1.1 f_{ij} P_i^{base} \quad (14)$$

$$j^* = \arg \max_j B_{ij}^{(2)} \quad (15)$$

3.5. Genetic representation and selection equations

The optimization model uses an assignment layer and an order layer. For N tasks, the assignment layer is a length- N vector $a = (a_1, \dots, a_N)$, where $a_i \in V \cup U$ gives the vehicle or UAV label assigned to task i . The order layer is a resource-indexed set of ordered lists $\Pi = \{\pi_j; j \in V \cup U\}$, where each non-empty π_j gives the service sequence of resource j . A candidate solution is infeasible when a task appears in more than one resource list, is absent from all lists, is inconsistent between a and Π , or violates UAV eligibility. The corresponding repair procedure is described in Section 4.

The evolutionary selection terms are defined by Eqs (16)–(18). Roulette-wheel sampling is performed after fitness shifting in Eq (16). Elitist preservation keeps the top E individuals as in Eq (17), and Eq (18) defines the generation-dependent crossover schedule.

$$\tilde{F}(X_i) = F(X_i) - \min(0, F_{min}) + \epsilon, P_i = \frac{\tilde{F}(X_i)}{\sum_{j=1}^N \tilde{F}(X_j)} \quad (16)$$

$$P^{(t+1)} = \left(P_{off}^{(t+1)} \setminus \text{Worst}_E(P_{off}^{(t+1)}) \right) \cup \text{Elite}_E(P^{(t)}) \quad (17)$$

$$p_c(g) = \begin{cases} p_{c,max}, & g \leq 0.3G, \\ \frac{p_{c,max} + p_{c,min}}{2}, & 0.3G < g < 0.7G, \\ p_{c,min}, & g \geq 0.7G. \end{cases} \quad (18)$$

The mutation and repair operators use the feasibility definitions above. Their procedural order, tie-breaking rule, and treatment of unassigned tasks are given in Algorithm 1 and Section 4.3.

3.6. Route optimization and collaborative objective

Given the repaired assignment, route sequencing is optimized independently for each resource by a TSP-based module as Eq (19), where T_{rem} is the set of unassigned or duplicated tasks after the auction stage, and d_{act} is the executed travel distance after the road/traffic adjustment. For resource j , the route cost and arrival time are Eq (20). Vehicles use the traffic factor, whereas UAVs do not. The final solution is evaluated by Eq (21). The normalized makespan and cost are computed by Eq (22). The obstacle-avoidance rate is Eq (23).

$$X^*, \{\pi_j\}_{j \in V \cup U} = GA(T_{rem}, V, U, UX) \quad (19)$$

$$Z_j = \sum_{r=0}^{m_j} d_{act}(t_r, t_{r+1}), Arr_r = Arr_{r-1} + S_{r-1} + \frac{d_{act}(r, -1r)}{v_j g_{tra}} \cdot 60 \quad (20)$$

$$F(S) = w_1 \tilde{C}_{max}(S) + w_2 \tilde{C}_{cost}(S) + w_3 R_{avoid}(S) + w_4 \rho(S) + w_5 F_{feas}(S) \quad (21)$$

$$\tilde{C}_{max}(S) = \frac{I}{I + C_{max}(S)/200}, \tilde{C}_{cost}(S) = \frac{I}{I + C_{cost}(S)/1000} \quad (22)$$

$$R_{avoid}(X) = \frac{\sum_{e \in E_V(X)} Q(e) + \sum_{a \in E_U(X)} Q(a)}{|E_V(X)| + |E_U(X)|} \quad (23)$$

The feasibility term $F_{feas}(S)$ records the constraint status after repair and final route evaluation. The implementation order of selection, elitism, crossover, mutation, and repair is specified in Section 4.

4. Algorithmic framework of IFAGA

IFAGA combines two hierarchical auction layers with a collaborative evolutionary optimization layer, as illustrated in Figure 1. The full procedure is summarized in Algorithm 1 before the auction stages are described in detail.

4.1. The first-round auction

The first-round auction implements the depot-level bidding model in Section 3.3. Each task sends its requirement profile to all depots, and only depots whose fuzzy fitness exceeds the first-round threshold submit valid bids. The highest valid bid selects the candidate service depot, which reduces the candidate set before execution-level resource competition.

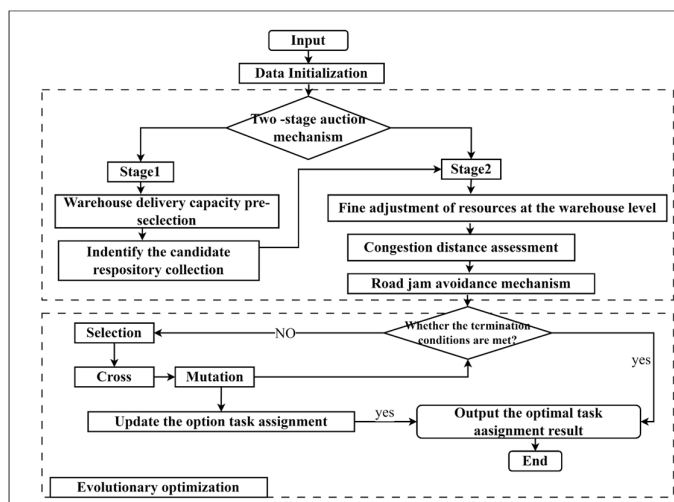


Figure 1. Overview of the designed IFAGA algorithm.

4.2. The second-round auction

In the second-round auction, only resources belonging to the winning depot compete for the corresponding task. The procedure applies the resource-level indicators in Section 3.4, excludes UAV-ineligible candidates from UAV bidding, and selects the highest valid second-round bid as the initial execution resource. This assignment initializes the evolutionary stage and is not treated as a final feasibility certificate.

The auction-generated assignment is then refined by collaborative evolutionary optimization. This stage updates route sequences, repairs duplicated or missing assignments, and rebalances workloads across heterogeneous resources before final route evaluation.

4.3. Complexity reduction and conflict handling

The proposed two-round design reduces computational burden by decomposing the global competition into two hierarchical stages. In a direct single-stage auction, each task would need to compete with all available vehicles and UAVs in the system. By contrast, this mechanism first filters tasks at the depot level and then restricts second-round competition to the resources managed by the winning depot. Let T denote the number of tasks, D is the number of depots, and R is the total number of resources. The candidate evaluation burden is therefore reduced from a global all-resource competition to a hierarchical process of approximately $O(|T| |D| + |T| r_d)$, where r_d is the average number of resources in a depot and typically satisfies $r_d \ll |R|$.

Within the overall experimental framework, the auction module serves as a front end that reduces complexity before the evolutionary search begins. The total computational cost of IFAGA can therefore be understood as the combined cost of hierarchical bidding, population-based search, and solution repair and evaluation. In practical settings, such as the present experiments, where the numbers of depots and resources are moderate, the two-round design remains tractable while preserving solution quality.

Conflict handling is also organized hierarchically. First, bidding conflicts among depots or resources are resolved locally by winner selection based on the maximum valid bid price, so that each

task accepts only one winning depot in the first round and one winning execution resource in the second round. Second, residual conflicts that may arise after the evolutionary search, such as duplicated assignments, unassigned tasks, or violations of capacity, time-window, and sortie-range constraints, are corrected by the repair stage before final route evaluation.

Algorithm 1. IFAGA for vehicle-UAV collaborative delivery.

Require: Task set Q , resource set $R = V \cup D$, initial auction output S_0 .
 Ensure: Best feasible solution S^* .

- 1: Encode each task with one resource label and store the ordered task lists by resource.
- 2: Initialize a population of size P from S_0 .
- 3: **for** $g = 1, \dots, G$ **do**
- 4: Select parents by elitism and roulette sampling.
- 5: Apply the generation-dependent crossover schedule and mutation.
- 6: Repair every offspring in the fixed order:
 - duplicate tasks are removed by first retained occurrence;
 - missing tasks are inserted into the highest-scoring feasible resource;
 - ties follow vehicle-first ascending resource labels;
 - if no feasible resource exists, the task remains unassigned and is penalized in feasibility evaluation.
- 7: Evaluate fitness, feasibility, route cost, and makespan.
- 8: Update the population by elitism.
- 9: **End for**
- 10: Select the best individual and optimize each resource route by the TSP-based ordering step.
- 11: **return** S^* .

5. Simulation experiments and result analysis

5.1. Experimental setup

The experiments used three depots, five heterogeneous ground vehicles, and five heterogeneous UAVs. The fixed benchmark used one 15-task instance within a study area spanning approximately 103.78°E to 104.60°E and 30.35°N to 30.90°N, with static task availability and traffic-obstacle disturbances. A separate event-based experiment evaluated staged order release and rolling-horizon re-optimization, and a scalability experiment evaluated randomly generated multi-instance task sets.

A hierarchical weighting scheme was used in the fuzzy evaluation process. The depot-level evaluation assigned weights of [0.40, 0.35, 0.25] to location proximity, capability matching, and time-window feasibility, respectively. In the vehicle round, the indicator order was path synergy, residual-capacity adequacy, economic-cost suitability, time-window slack, and traffic index, with weights [0.25, 0.20, 0.20, 0.18, 0.17]. In the UAV round, the indicator order was endurance adequacy, airspace-risk suitability, time urgency, task matching, and depot-context dependency, with weights [0.25, 0.18, 0.17, 0.22, 0.18]. The last UAV indicator represented land-use and operating context rather than mother-ship coupling.

To reflect dynamic traffic disturbances, four categories of road obstacles were introduced, namely construction, accident, control, and congestion. Their occurrence probabilities and influence ranges were differentiated according to disturbance severity, and most obstacles were generated on the road network, while a smaller proportion was placed in high frequency congestion areas, and the remainder

was introduced as random disturbances. This obstacle design preserved spatial heterogeneity and disturbance diversity in the benchmark environment.

The proposed method was compared with Adaptive Large Neighborhood Search (ALNS), Classical Auction with Dynamic Programming (Classical-DP), NSGA-II, Variable Neighborhood Search (VNS), and a discrete particle swarm optimization baseline (DPSO). Within each matched run, all methods used the same task set, obstacle realization, stopping budget, repair interface, route sequencing, and final evaluation procedure. The algorithm-specific settings were as follows: ALNS used 100 iterations, initial temperature 50, and cooling rate 0.90; NSGA-II used population size 40, 60 generations, non-dominated sorting, crowding-distance selection, and crossover rate 0.85; VNS used 120 iterations with reassign, swap, and route-move neighborhoods; Classical-DP used classical auction assignment followed by dynamic-programming route sequencing; and DPSO used 40 particles, 60 iterations, cognitive and social rates of 0.35 and 0.45, respectively, and mutation rate 0.20.

The experiments were organized in three settings. First, the main comparative analysis used the fixed 15-task benchmark, with 20 independent runs per algorithm reported as mean $\pm$ standard deviation. Second, the event-based rolling-horizon experiment released the 15 tasks at 0, 60, and 120 min under two deterministic patterns, uniform release with 5 / 5 / 5 tasks and burst release with 10 / 3 / 2 tasks. At each release time, IFAGA optimized only released and unfinished tasks, while tasks completed before the next release were frozen. Each condition and release pattern was evaluated in 20 paired runs using matched deterministic seeds. Third, the scalability study used randomly generated multi-instance task sets, with 10 independent instances per task scale and three repeated runs per instance before aggregation. The dynamic experiment reported completion rate, late-task rate, total lateness, dynamic makespan, total cost, obstacle-avoidance rate, re-optimization count, and re-optimization runtime. Paired differences were evaluated with two-sided paired t-tests and 95% confidence intervals, with Holm correction across the five reported dynamic metrics within each pattern; wilcoxon signed-rank tests were retained as a non-parametric sensitivity analysis.

The comparison was therefore a controlled benchmark under matched algorithmic settings. The reported differences reflected performance under the stated settings rather than the best attainable result after exhaustive per-baseline retuning.

The table of delivery units and their attribute are shown in Table 2.

Table 2. Bidding prices of the first round of auction.

Warehouse	Resource Type (V/D)	Code	Capacity (kg)	Speed(km/h)	Endurance (km)
d01	V/D	V01	960	20	200
		D01	20	15	15
		V04	500	40	180
d02	V/D	V02	150	40	200
		D02	20	20	18
		D03	20	20	18
		V03	800	20	180
d03	V/D	V05	800	30	130
		D04	50	50	18
		D05	50	50	18

The fixed 15-task benchmark was held constant across algorithms. Task coordinates were generated within 103.78°E to 104.60°E and 30.35°N to 30.90°N. Task types were sampled from urgent

delivery, precision delivery, bulk transport, remote delivery, and sensitive transport. The corresponding capability requirements used bounded resource-to-demand ratios for speed, precision, capacity, range, stealth, reliability, agility, and endurance. Task weights, time windows, priorities, base prices, and UAV eligibility were generated by the same deterministic rule. No separate service-time variable was used; service feasibility was represented through time windows, travel time, arrival time, and repair feasibility. The complete fixed benchmark specification is provided in Appendix A.

Random seed 42 was used as the base seed for the fixed benchmark construction. The 20 repeated runs used seeds 42 to 61, while the scalability experiment used instance seeds defined as $42 + 1000n + s$ for task scale n and instance index s , followed by algorithm-specific run seeds. The genetic stage used population size 50, 150 generations, $p_{c_{\max}} = 0.90$, $p_{c_{\min}} = 0.50$, mutation rate 0.10, and elitist retention of four individuals.

5.2. Analysis and explanation of experimental results

Subject to the constraints specified in Section 3, the experiments first evaluated depot-task compatibility and then resource-task compatibility for vehicles and UAVs. The fuzzy assessment quantified how well each candidate depot or execution resource matched the requirements of each task.

To illustrate how the algorithm operates in practice, we present a representative case that traces how IFAGA turns hierarchical bidding decisions into a coordinated delivery plan.

In the first round of fuzzy bidding, warehouses submitted bids for tasks, which were then allocated to specific warehouse locations. The bidding prices are illustrated in Figure 2. In the bidding-price subplot, warehouses enclosed by red solid lines represent the winning warehouses, while the gray dashed lines denote the average bidding price of all warehouses in this auction round.

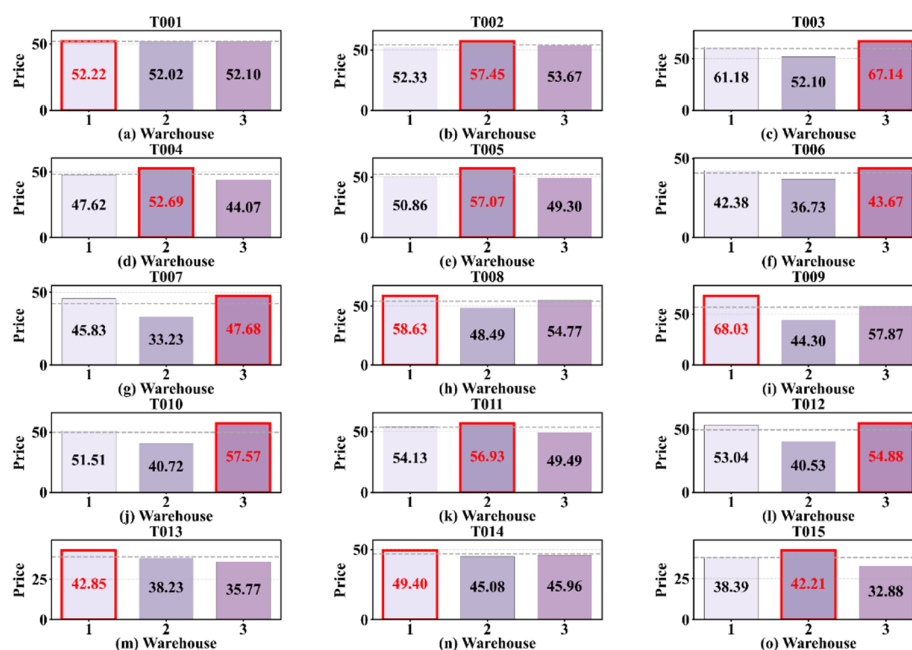


Figure 2. Bidding prices of the first round of auction.

In the resource-level optimization phase, the fitness between resources and tasks was calculated to complete the second round of bidding. The winning bidders and their winning prices for the resource-task pairs are illustrated in Figure 3. Green represents vehicles, and light blue represents UAVs. These bidding results described the candidate assignments before the genetic repair and route-optimization stage.

For UAVs, feasibility was enforced at the single-sortie round-trip level. Therefore, cumulative executed distance may have exceeded the length of any one sortie, while the route remained feasible only if the maximum single-sortie round-trip distance did not exceed the rated range.

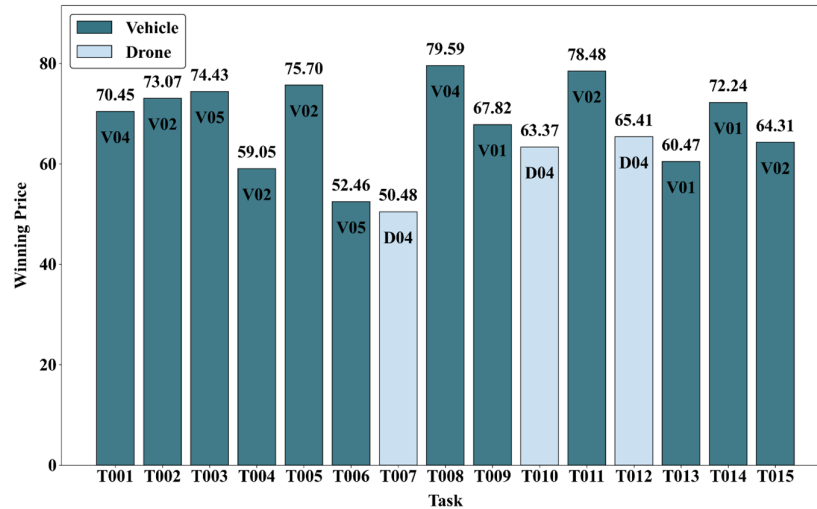


Figure 3. Results of individual resource bidding in the second round of auction.

Table 3. Task-agent allocation results.

Agent	Type	Number of Tasks	Total Distance (km)	Total Time (min)
V01	V	3	49.24	115.3
V02	V	3	89.28	113.24
V05	V	3	77.54	109.58
V04	V	2	61.33	97.79
V03	V	1	28.89	56.81
D04	D	3	5.16	6.19
Total	–	15	311.44	498.91

After the two bidding rounds, the improved genetic algorithm repaired the initial allocation and refined the routes globally. The representative final solution in Table 3 remained vehicle-dominant, with selective UAV support. Five vehicle agents and one UAV agent appeared in the final execution set. V01, V02, and V05 each executed three tasks, V04 executed two tasks, V03 executed one task, and UAV D04 executed three tasks. The cumulative executed time across the active agents was 498.91 min, and the total travel distance was 311.44 km.

Figure 4 visualizes the final optimized routes and shows the same vehicle-led pattern with selective UAV assistance. V01 serves T009, T0013, and T001; V02 serves T011, T005, and T002; V05

serves T003, T014, and T008; V04 serves T006; V03 serves T004 and T015; and D04 serves T010, T012, and T007. This result indicates a vehicle-dominant execution pattern with selective UAV assistance. The UAV was used only for tasks whose payload requirements, drone eligibility, and single-sortie round-trip range remained feasible.

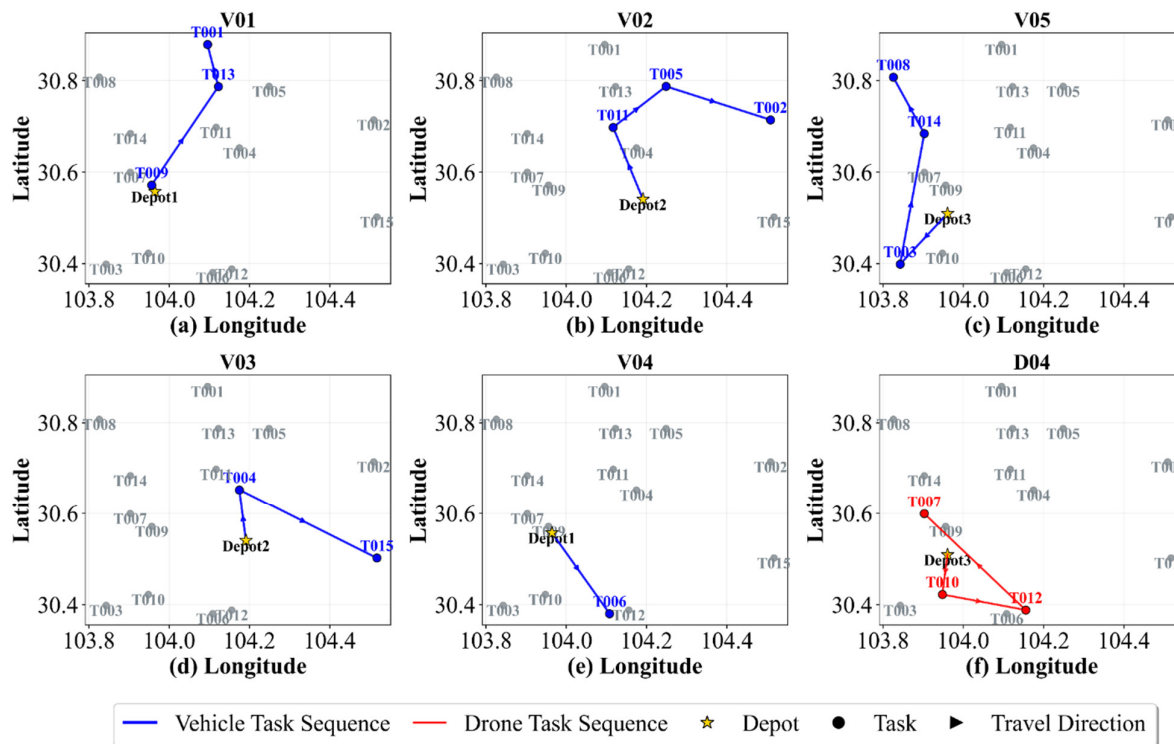


Figure 4. Representative final allocation after feasibility repair and route optimization.

5.2.1. Comprehensive performance analysis results

All comparison algorithms were run under identical task instances, the same random seed rules, and the same road-block generation outcomes, and they shared a consistent procedure for repair, route generation, and result evaluation. The main comparative experiments were conducted on a fixed 15-task benchmark, with each algorithm independently repeated 20 times, and the results were reported as mean and standard deviation in Table 4, Table 5, and Figure 5. In Table 5, positive values indicate improvement for the corresponding metric, whereas negative values indicate that IFAGA was worse than the baseline for that metric.

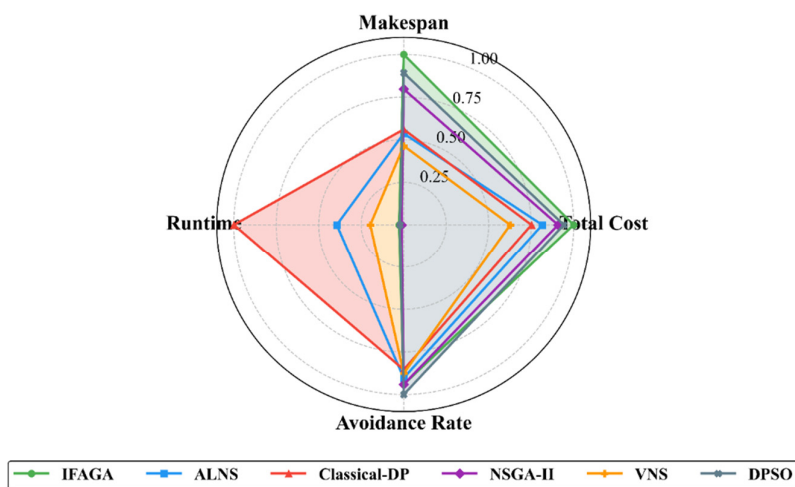
IFAGA achieved a mean makespan of 122.37 min, a mean total cost of 363.11, an obstacle-avoidance rate of 0.913, and a runtime of 1.80 s. On the fixed 15-task benchmark, IFAGA reduced makespan relative to ALNS, Classical-DP, NSGA-II, VNS, and DPSO by 46.38%, 43.94%, 20.32%, 53.58%, and 10.79%, respectively. Runtime remained higher than ALNS, Classical-DP, and VNS, but lower than NSGA-II and DPSO.

Table 4. Performance index comparison of five algorithms.

Algorithm	Makespan	Cost	Obstacle Avoidance Rate	Runtime
IFAGA	122.366 ± 13.831	363.109± 32.714	0.913 ± 0.024	1.802 ± 0.152
ALNS	228.222 ± 32.160	445.285± 56.628	0.877 ± 0.038	0.126 ± 0.015
Classical DP	218.290 ± 31.247	482.981± 39.436	0.829 ± 0.046	0.050 ± 0.008
NSGA-II	153.577 ± 28.140	399.852± 35.460	0.914 ± 0.028	4.038 ± 0.426
VNS	263.611 ± 46.695	579.280± 56.080	0.851 ± 0.048	0.252 ± 0.032
DPSO	137.170 ± 24.460	385.348± 39.796	0.971 ± 0.024	2.279 ± 0.242

Table 5. Relative improvement rate of IFAGA algorithm (%).

Algorithm	Makespan	Cost	Obstacle Avoidance Rate	Runtime
ALNS	46.38%	18.45%	4.13%	−1330.79%
Classical DP	43.94%	24.82%	10.09%	−3529.87%
NSGA-II	20.32%	9.19%	−0.04%	55.38%
VNS	53.58%	37.32%	7.33%	−614.53%
DPSO	10.79%	5.77%	−5.95%	20.94%

**Figure 5.** Radar chart for multi-algorithm performance comparison.

The cost and obstacle-avoidance results showed the same trade-off pattern. IFAGA reduced mean cost relative to ALNS, Classical-DP, NSGA-II, VNS, and DPSO by 18.45%, 24.82%, 9.19%, 37.32%, and 5.77%, respectively. Its obstacle-avoidance rate was higher than those of ALNS, Classical-DP, and VNS, nearly identical to NSGA-II, and lower than DPSO, which achieved 0.971 compared with 0.913 for IFAGA. These results support that IFAGA improved makespan and cost under the tested conditions, whereas DPSO provided the strongest obstacle-avoidance result in this comparison.

Against ALNS, IFAGA achieved a lower makespan, reducing the average completion time from 228.222 min to 122.366 min. IFAGA also reduced mean total cost from 445.285 to 363.109, while the obstacle-avoidance rate increased from 0.877 to 0.913. Runtime increased from 0.126 s to 1.802 s.

A similar pattern was observed in the comparison with Classical-DP. IFAGA reduced the average makespan from 218.290 min to 122.366 min and lowered mean total cost from 482.981 to 363.109. The obstacle-avoidance rate also improved from 0.829 to 0.913. Runtime increased from 0.050 s to 1.802 s.

The comparison with NSGA-II showed a closer solution-quality result than the other baselines. IFAGA had lower mean makespan and cost, while the obstacle-avoidance rates were nearly identical. Runtime also differed: IFAGA required 1.802 s, whereas NSGA-II required 4.038 s.

Compared with VNS, IFAGA reduced average makespan from 263.611 min to 122.366 min, decreased total cost from 579.280 to 363.109, and increased the obstacle-avoidance rate from 0.851 to 0.913. Runtime increased from 0.252 s to 1.802 s.

Taken together, the four metrics indicated that IFAGA primarily improved makespan under the fixed benchmark while retaining competitive cost and obstacle-avoidance performance. This pattern was accompanied by higher runtime than ALNS, Classical-DP, and VNS, but lower runtime than NSGA-II and DPSO.

5.2.2. Analysis of statistical test results

To assess the stability of the pairwise comparisons, we report the paired t-test, the Wilcoxon signed-rank test, and Cohen's d for IFAGA against each baseline in Table 6 and Table 7. These statistics, as also illustrated in Figure 6, show whether the descriptive trends in the benchmark table were consistent across parametric and nonparametric tests and provide a simple effect-size reference.

The paired t-tests and Wilcoxon tests provided consistent evidence for lower IFAGA makespan than every baseline. The t-test p-values were below 0.001 for ALNS, Classical-DP, NSGA-II, and VNS, and 0.0059 for DPSO. The corresponding Wilcoxon p-values were below 0.005 for all five comparisons. The effect sizes were large against the four conventional baselines and moderate against DPSO.

Cost differences were significant under both tests for ALNS, Classical-DP, NSGA-II, and VNS. The DPSO comparison was less stable, with $p = 0.0481$ under the paired t-test and $p = 0.0696$ under the Wilcoxon test. Therefore, the data do not support a robust cost advantage over DPSO.

Table 6. Paired t test (t-statistic, p-value, Cohen's d).

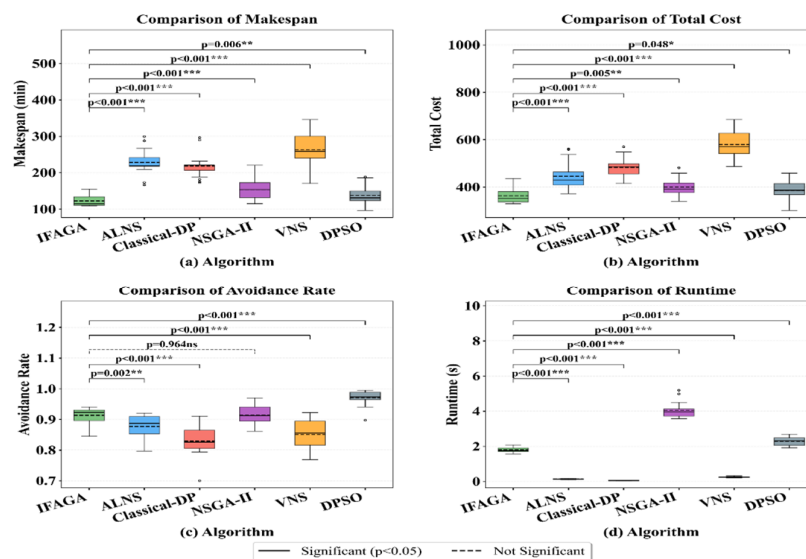
Metric	ALNS	Classical DP	NSGA-II	VNS	DPSO
Makespan	-13.01	-15.65	-4.53	-13.27	-3.10
	(6.5e - 11)	(2.6e - 12)	(2.3e - 4)	(4.7e - 11)	(0.0059)
	4.28	3.97	1.41	4.10	0.69
Cost	-5.33	-10.19	-3.13	-16.55	-2.11
	(3.8e - 5)	(3.9e - 9)	(0.0055)	(9.7e - 13)	(0.0481)
	1.78	3.31	1.08	4.71	0.47
Avoidance Rate	3.49	6.97	-0.05	6.28	-10.41
	(0.0025)	(1.2e - 6)	(0.964)	(5.0e - 6)	(<0.001)
	1.14	2.29	-0.02	1.64	-2.33
Runtime	51.74	53.10	-26.36	50.28	-9.30
	(<0.001)	(<0.001)	(<0.001)	(<0.001)	(<0.001)
	15.55	-16.32	6.99	-14.14	2.08

*Note: * $p < 0.05$, ** $p < 0.01$, *** $p < 0.001$, $p > 0.05$ ns.

Table 7. Paired wilcoxon test (wilcoxon statistic, p-value).

Metric	ALNS	Classical DP	NSGA-II	VNS	DPSO
Makespan	0 ($1.9e-6$)	0 ($1.9e-6$)	11 (0.00072)	0 ($1.9e-6$)	32 (0.0049)
Cost	6 ($2.7e-5$)	0 ($1.9e-6$)	35 (0.0073)	0 ($1.9e-6$)	56 (0.0696)
Avoidance Rate	29 (0.0032)	8 ($4.8e-5$)	104 (0.985)	3 ($9.5e-6$)	0 (<0.001)
Runtime	0 ($1.9e-6$)	0 ($1.9e-6$)	0 ($1.9e-6$)	0 ($1.9e-6$)	0 (<0.001)

*Note: * $p < 0.05$, ** $p < 0.01$, *** $p < 0.001$, $p > 0.05$ ns.

**Figure 6.** Boxplot of statistical-test results for IFAGA versus five baseline algorithms.

Obstacle-avoidance differences were significant for ALNS, Classical-DP, VNS, and DPSO, but not for NSGA-II. The direction against DPSO was unfavorable to IFAGA, consistent with the higher DPSO avoidance rate in Table 4. Runtime differences were significant for all five baselines, with IFAGA slower than ALNS, Classical-DP, and VNS, but faster than NSGA-II and DPSO.

The effect sizes in Table 6 indicate that the largest differences occurred for several conventional-baseline and runtime comparisons. The DPSO comparisons showed smaller effects for makespan and cost and an opposite direction for obstacle avoidance. These results support a metric-specific interpretation rather than uniform dominance across all metrics.

The statistical tests supported the same balanced reading of IFAGA. The method was consistently better than ALNS, Classical-DP, and VNS on makespan, while NSGA-II remained slightly stronger on some solution-quality metrics but slower to run. Overall, the results suggested a trade-off between solution quality and computational burden rather than uniform dominance across all metrics.

5.3. Dynamic order release and rolling-horizon validation.

The dynamic-order experiment compared rolling re-optimization with a no-reoptimization control after orders were released at 0, 60, and 120 min. In the rolling setting, each decision epoch optimized

only released and unfinished tasks. The control kept the initial plan and inserted later orders greedily. Resource location, availability time, completed-task state, and interpolated in-transit position were carried forward between epochs. Two release patterns were tested with 20 matched runs per condition using the same seeds, and all 80 runs completed the 15 tasks.

Under uniform release, rolling re-optimization achieved a makespan of 252.79 ± 26.74 min, total cost of 277.54 ± 43.56 , and obstacle-avoidance rate of 0.931 ± 0.045 . The corresponding control values were 276.25 ± 51.08 min, 363.12 ± 65.01 , and 0.881 ± 0.075 . Under burst release, rolling re-optimization yielded 258.40 ± 25.15 min, 311.32 ± 71.07 , and 0.921 ± 0.052 , compared with 267.53 ± 25.14 min, 363.82 ± 84.44 , and 0.905 ± 0.075 for the control.

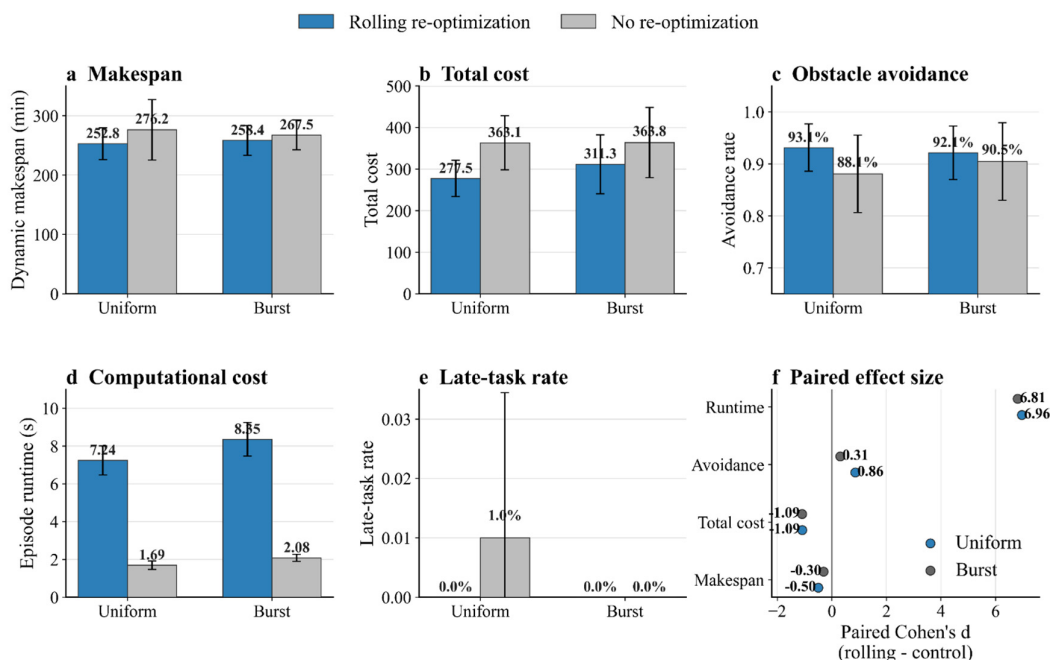


Figure 7. Dynamic order release under rolling re-optimization and control.

After Holm correction, paired differences indicated lower total cost under rolling re-optimization in both release patterns. The mean difference was -85.59 for uniform release, with a 95% CI from -122.45 to -48.72 , and -52.49 for burst release, with a 95% CI from -74.99 to -30.00 . The uniform pattern also had a higher avoidance rate, with a mean difference of 0.0506 and a 95% CI from 0.0229 to 0.0783 . The burst-pattern avoidance difference and the makespan differences were not supported after correction. Rolling re-optimization required three optimization cycles and longer episode runtimes than the control, with 7.24 ± 0.77 s versus 1.69 ± 0.22 s under uniform release and 8.35 ± 0.89 s versus 2.08 ± 0.18 s under burst release. These results indicated that rolling re-optimization reduced total cost across the tested staged-release settings, improved obstacle avoidance under uniform release, and increased computational time. The transferred in-transit state was represented by geometric interpolation between consecutive task locations.

The corresponding group-level summaries and paired effect sizes are shown in Figure 7. Panels a-e show the mean $\pm$ s.d. across 20 runs per condition and release pattern for dynamic makespan, total cost, obstacle-avoidance rate, episode runtime, and late-task rate, respectively. Uniform release denotes

5 / 5 / 5 task arrivals and burst release denotes 10 / 3 / 2 arrivals at 0, 60, and 120 min. Panel f reports paired Cohen's d for rolling re-optimization relative to the no-reoptimization control within each release pattern. The paired tests used two-sided paired t -tests with 95% confidence intervals and Holm correction across the five dynamic metrics; Wilcoxon signed-rank tests were used as a non-parametric sensitivity analysis.

5.4. Impact of task scale

Table 8. Summary of scalability test results.

algorithm	metrics	Task scale			
		15	30	50	80
IFAGA	makespan	140.6763	254.0831	529.7998	995.9413
	Total cost	399.0878	742.0094	1430.309	2802.821
	avoidance rate	0.898231	0.895781	0.876323	0.856995
	runtime	1.825686	2.765419	4.351361	7.939909
ALNS	makespan	274.1129	541.9919	702.1102	1209.311
	total cost	456.3329	875.034	1444.746	2848.04
	avoidance rate	0.871071	0.867971	0.869043	0.852316
	runtime	0.176489	0.391539	1.002158	2.842923
Classical-DP	makespan	247.9313	399.3534	834.9323	1376.325
	total cost	476.5808	851.106	1554.5	2734.666
	avoidance rate	0.847306	0.87527	0.860674	0.85799
	runtime	0.116823	0.199205	0.402241	0.812946
NSGAI	makespan	171.5139	380.8006	759.5228	1628.265
	total cost	412.9926	874.3215	1590.122	3188.649
	avoidance rate	0.893892	0.890168	0.86837	0.841527
	runtime	4.087463	5.920907	8.37047	14.56017
VNS	makespan	280.5937	458.4075	854.5957	1439.425
	total cost	614.4379	1136.336	2013.993	3793.195
	avoidance rate	0.77677	0.828737	0.82311	0.790479
	runtime	0.297914	0.72162	1.228911	1.773396

To evaluate the adaptability of the algorithms across problem scales, we constructed random instances with four task sizes, namely 15, 30, 50, and 80, and generated 10 independent instances for each size. Each instance was run three times for each algorithm. The scalability results were aggregated in a two-stage manner: first within each instance by averaging repeated runs, and then across instances by summarizing the 10 instance-level means. This procedure prevented random fluctuations within the same instance from being conflated with the inherent variability across instances. The results are presented in Table 8 and Figure 8.

It should be noted that the 15-task results reported in Table 4 and Table 8 were obtained under different experimental settings. Table 4 is based on a fixed 15-task benchmark instance for controlled algorithm comparison, whereas Table 8 summarizes scalability results over randomly generated instances at different task scales.

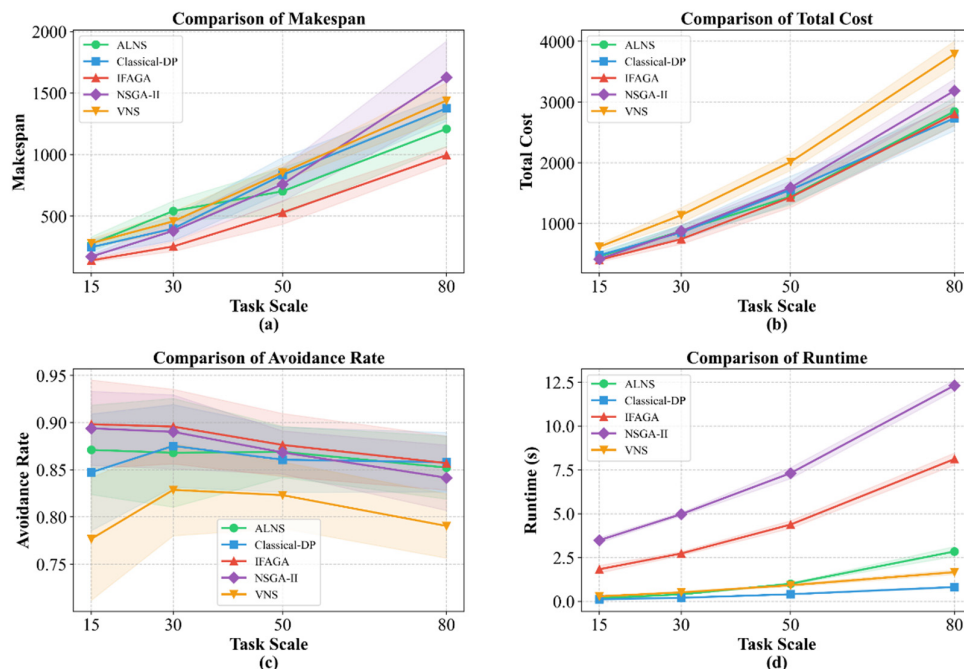


Figure 8. Scalability test comparison.

At 15 tasks, IFAGA achieved the best results in average makespan, total cost, and avoidance rate, with values of 140.68 min, 399.09, and 0.8982, respectively. NSGA-II ranked second, with corresponding values of 171.51 min, 412.99, and 0.8939. At 30 tasks, IFAGA performed best on makespan, total cost, and avoidance rate, yielding 254.08 min, 742.01, and 0.8958 for each metric, while NSGA-II and Classical-DP followed in makespan with 380.80 min and 399.35 min, respectively. At 50 tasks, IFAGA again ranked first in makespan, total cost, and avoidance rate, with results of 529.80 min, 1430.31, and 0.8763, respectively.

Moreover, ALNS achieved the closest performance in total cost with a value of 1444.75 and ranked second in makespan with 702.11 min. At 80 tasks, IFAGA still obtained the lowest makespan at 995.94 min. In comparison, the makespan of ALNS, Classical-DP, VNS, and NSGA-II was 1209.31 min, 1376.32 min, 1439.42 min, and 1628.27 min, respectively. At this scale, Classical-DP attained the lowest total cost and the highest avoidance rate, with values of 2734.67 and 0.8580 for each metric, while IFAGA followed closely with 2802.82 and 0.8570, respectively.

For runtime, Classical-DP was the fastest method across all task scales, whereas NSGA-II consistently had the highest computation time throughout the experiments. The runtime of IFAGA increased with task scale, rising from 1.83 s at 15 tasks to 7.94 s at 80 tasks, but it maintained the best makespan at all tested scales while showing strong competitiveness in total cost and avoidance performance, especially in medium-to-large-scale task scenarios.

The scalability results suggest that runtime increases with task scale, as expected, while solution quality remains relatively stable and makespan performance stays favorable. In medium-to-large-scale scenarios, the algorithm continues to deliver feasible solutions with competitive makespan outcomes, which supports its adaptability as problem size grows.

5.5. Parameter sensitivity analysis

A parameter sensitivity analysis was conducted for population size, maximum generation number, mutation rate, maximum crossover rate, and elitist retention. Each value was evaluated in 20 independent runs on the 15-task setting using deterministic seed construction. The tested ranges produced mean makespans of 133.08–142.79 min for population sizes of 30–80, 136.07–151.65 min for maximum generations of 50–200, 132.42–140.89 min for mutation rates of 0.05–0.20, 125.59–140.39 min for maximum crossover rates of 0.75–0.95, and 129.93–141.44 min for elite sizes of 2–8. Runtime increased with population size and generation count, from 1.36 to 2.16 s across the population-size range and from 1.17 to 1.66 s across the generation range. The selected configuration therefore represented a practical compromise among solution quality, obstacle avoidance, and computation time.

Table 9. Membership-weight sensitivity analysis for the fuzzy-evaluation weights.

Weight group	Setting	Makespan	Cost	Obstacle avoidance rate	Composite fitness
Depot	Baseline	139.58 ± 16.73	409.57 ± 47.39	0.877 ± 0.074	0.7612 ± 0.0256
Depot	Concentrated	139.73 ± 15.50	401.33 ± 36.19	0.887 ± 0.056	0.7561 ± 0.0245
Depot	Flattened	144.05 ± 20.21	404.69 ± 59.45	0.878 ± 0.075	0.7529 ± 0.0263
Vehicle	Baseline	141.18 ± 21.45	417.89 ± 66.31	0.874 ± 0.057	0.7546 ± 0.0288
Vehicle	Concentrated	142.40 ± 11.88	422.60 ± 35.51	0.885 ± 0.041	0.7571 ± 0.0157
Vehicle	Flattened	137.83 ± 15.49	410.01 ± 49.70	0.900 ± 0.041	0.7644 ± 0.0226
UAV	Baseline	135.36 ± 14.58	406.36 ± 45.92	0.892 ± 0.039	0.7653 ± 0.0179
UAV	Concentrated	142.25 ± 17.26	418.82 ± 56.38	0.886 ± 0.062	0.7578 ± 0.0234
UAV	Flattened	139.89 ± 18.37	408.37 ± 49.42	0.892 ± 0.040	0.7631 ± 0.0169

Table 10. Sensitivity of the composite objective to the makespan / cost normalization scales in Eq (21).

Scale pair	Makespan (min)	Cost	Composite fitness
100 / 500	139.91 ± 17.52	409.13 ± 56.97	0.6634 ± 0.0276
150 / 750	140.80 ± 17.96	414.55 ± 66.20	0.7144 ± 0.0226
200 / 1000	145.37 ± 22.68	419.70 ± 52.57	0.7585 ± 0.0204
300 / 1500	134.92 ± 16.72	399.02 ± 44.48	0.8113 ± 0.0342
400 / 2000	139.12 ± 18.86	406.38 ± 52.12	0.8449 ± 0.0225

The fuzzy-evaluation weights were further examined by a one-at-a-time perturbation analysis on the fixed 15-task setting. For the depot, vehicle, and UAV weight vectors, two perturbed settings were generated by transferring 0.05 between the largest and smallest entries in opposite directions, followed by renormalization. The baseline, concentrated, and flattened settings were each evaluated in 20 independent runs using the same evaluation pipeline. The results are summarized in Table 9.

The perturbation results showed metric-specific sensitivity rather than invariance to the manually specified weights. Flattening the vehicle weights produced the lowest mean makespan and the highest

mean obstacle-avoidance rate among the vehicle-weight settings, whereas concentrating the UAV weights increased the mean makespan and reduced composite fitness. The default weights therefore provided a feasible configuration for the reported experiments.

The scale factors in Eq (21) were also examined because they determined the numerical balance between normalized makespan and normalized cost in the composite objective. Five proportional makespan/cost scale pairs were evaluated for IFAGA on the fixed 15-task setting, with 20 independent runs for each pair. As shown in Table 10, the composite-fitness mean increased from 0.6634 under the 100 / 500 scale pair to 0.8449 under the 400 / 2000 scale pair. Across the five settings, the mean makespan ranged from 134.92 to 145.37 min and the mean cost ranged from 399.02 to 419.70. These results indicated that the scale factors affected the composite-fitness level. This analysis was conducted for IFAGA and did not establish that the full cross-algorithm ranking was invariant to alternative scale choices.

6. Conclusion

Under the tested static task sets and matched evaluation procedure, IFAGA achieved the lowest mean makespan across the reported task scales, while cost, obstacle avoidance, feasibility, and runtime varied across baselines. In the event-based experiment, rolling re-optimization completed all tasks in both release patterns and reduced total cost relative to the matched no-reoptimization control, while the avoidance benefit was pattern-dependent and the computational cost was higher. These results support an integration contribution that balances assignment screening, repair, and route refinement under the tested conditions.

Additionally, the event-based results were interpreted as a staged-release stress test rather than a full dynamic traffic-control model.

Several limitations should be acknowledged. First, the dynamic validation uses only two deterministic three-stage release patterns and a no-reoptimization control, so it does not establish robustness to broader arrival processes, demand uncertainty, or rolling-horizon policies with variable decision intervals. Second, resource position during an unfinished movement is transferred by geometric interpolation between consecutive task locations, which approximates execution state but does not reconstruct a road-network trajectory. Third, traffic and airspace effects are represented by simplified factors rather than predictive traffic or explicit airspace-control models. Fourth, the weight analysis quantifies one-at-a-time perturbations of the three indicator groups, but it does not identify a statistically optimal weight vector or explore joint perturbations. Fifth, the 200 and 1000 normalization scales produce different composite-fitness levels across the tested scale pairs. The scale analysis was performed for IFAGA and therefore does not establish invariance of the full cross-algorithm ranking. Sixth, the fixed-benchmark results depend on the specified task and depot attributes and deterministic generation rule, so transfer to unobserved input distributions remains to be established. Seventh, the fixed-benchmark comparison includes a swarm-intelligence comparator (DPSO), but does not include a learned multi-agent reinforcement-learning policy, which would require a separate training, generalization, and deployment protocol. In future work, we will address richer dynamic task arrival, predictive traffic modeling, explicit airspace constraints, joint weight calibration, objective-scale selection, and a broader set of recent vehicle-UAV baselines.

Author contributions

Conceptualization, J.C.P and W.D.J.; Methodology, J.C.P.; Software, J.C.P.; Validation, J.C.P.; Formal Analysis, J.C.P.; Investigation, J.C.P. and C.C.; Resources, W.D.J.; Data Curation, J.C.P.; Writing – Original Draft Preparation, J.C.P. and C.C.; Writing – Review & Editing, W.D.J.; Visualization, J.C.P.; Supervision, W.D.J.; Project Administration, W.D.J.; Funding Acquisition, W.D.J.

Use of Generative-AI tools declaration

The author(s) declare that Artificial Intelligence (AI) tools were used in the writing of this article. The AI tool used was ChatGPT (OpenAI). It was used solely for language polishing and editing, including improvements in grammar, spelling, punctuation, clarity of expression, and readability. All AI-assisted revisions were reviewed and approved by the author(s), who take full responsibility for the final content of the manuscript.

Acknowledgments

The authors would like to express their deepest gratitude for the assistance provided by all co-authors during the writing of the paper. In addition, the authors would like to express their gratitude to many senior scholars, whose related research has laid a solid foundation for this study and is listed as references in this paper.

Conflict of interest

The authors declare that there are no conflicts of interest.

References

1. A. Arenas–Vasco, J. C. Rivera, M. G. Baldoquín, New formulation and valid inequalities for a periodic capacitated vehicle routing problem with multiple depots, heterogeneous fleet, and hard time-windows, *PLOS One*, **20** (2025), e0335389. <https://doi.org/10.1371/journal.pone.0335389>
2. X. Cao, Z. Da, X. Wang, Order fulfillment planning with crowdsourcing integration, *Sci. Rep.*, **15** (2025), 27271. <https://doi.org/10.1038/s41598-025-11061-3>
3. M. A. Sebok, H. G. Tanner, Cooperative planning for physically interacting heterogeneous robots, *Front. Robot. AI*, **11** (2024), 1172105. <https://doi.org/10.3389/frobt.2024.1172105>
4. P. V. Ghatge, H. K. Pati, Collaborative distributed communication in heterogeneous environments: A comprehensive survey, *J. Netw. Comput. Appl.*, **61** (2016), 1–20. <https://doi.org/10.1016/j.jnca.2015.10.006>
5. J. Abusalama, S. Razali, Y. H. Choo, An enhanced approach for solving winner determination problem in reverse combinatorial auctions, *Indones. J. Electr. Eng. Comput. Sci.*, **28** (2022), 934–945. <https://doi.org/10.11591/ijeecs.v28.i2.pp934-945>
6. M. De Ryck, D. Pissort, T. Holvoet, E. Demeester, Decentral task allocation for industrial AGV-systems with routing constraints, *J. Manuf. Syst.*, **62** (2022), 135–144. <https://doi.org/10.1016/j.jmsy.2021.11.012>

7. R. Liu, H. S. Shin, B. Yan, A. Tsourdos, An auction-based coordination strategy for task-constrained multi-agent stochastic planning with submodular rewards, 2022. Available from: <https://arxiv.org/abs/2212.14624>
8. F. Quinton, C. Grand, C. Lesire, Communication-preserving bids in market-based task allocation, 2022 IEEE/RSJ IROS, (2022), 13708–13713. <https://doi.org/10.1109/IROS47612.2022.9981699>
9. H. B. Ozturk, N. B. Yavas, Z. Bingul, Adaptive market-based dynamic task allocation under environmental uncertainty, *Proc. 15th SIMULTECH*, **1** (2025), 70–80. <https://doi.org/10.5220/0013522600003970>
10. Y. Wang, H. Li, Y. Yao, An adaptive distributed auction algorithm and its application to multi-AUV task assignment, *Sci. China Technol. Sci.*, **66** (2023), 1235–1244. <https://doi.org/10.1007/s11431-022-2302-6>
11. M. Alshaboti, U. Baroudi, Multi-robot task allocation system: Fuzzy auction-based and adaptive multi-threshold approaches, *SN Comput. Sci.*, **2** (2021), 87. <https://doi.org/10.1007/s42979-021-00479-x>
12. F. Tao, W. F. Li, An improved auction algorithm for task allocation in RMFS, *Logist. Technol.*, **41** (2022), 63–68.
13. N. Fazal, M. T. Khan, S. Anwar, J. Iqbal, S. Khan, Task allocation in multi-robot system using resource sharing with dynamic threshold approach, *PLOS One*, **17** (2022), e0267982. <https://doi.org/10.1371/journal.pone.0267982>
14. H. Wu, H. Shang, Potential game for dynamic task allocation in multi-agent system, *ISA Trans.*, **102** (2020), 208–220. <https://doi.org/10.1016/j.isatra.2020.03.004>
15. Q. Li, T. W. Fan, L. S. Kei, Z. Li, Scalable and energy-efficient task allocation in industry 4.0: Leveraging distributed auction and IBPSO, *PLOS One*, **20** (2025), e0314347. <https://doi.org/10.1371/journal.pone.0314347>
16. H. L. Choi, L. Brunet, J. P. How, Consensus-based decentralized auctions for robust task allocation, *IEEE Trans. Robot.*, **25** (2009), 912–926. <https://doi.org/10.1109/TRO.2009.2022423>
17. G. Wang, Q. S. Chen, Y. Q. Du, X. W. Quan, Multi-agent single-machine scheduling problem based on combinatorial auction, *Comput. Integr. Manuf. Syst.*, **19** (2013), 106–113.
18. L. Wang, M. Liu, M. Q. H. Meng, A hierarchical auction-based mechanism for real-time resource allocation in cloud robotic systems, *IEEE Trans. Cybern.*, **47** (2017), 473–484. <https://doi.org/10.1109/TCYB.2016.2519525>
19. Y. L. Ren, X. Y. Zou, X. H. Zhang, Bidding game model for power generation companies based on first-price sealed auction, *J. Syst. Eng.*, **18** (2003), 248–254.
20. J. M. Garibaldi, T. Ozen, Uncertain fuzzy reasoning: A case study in modelling expert decision making, *IEEE Trans. Fuzzy Syst.*, **15** (2007), 16–30. <https://doi.org/10.1109/TFUZZ.2006.889755>
21. D. R. Wu, Z. G. Zeng, H. Mo, F. Y. Wang, Interval type-2 fuzzy sets and systems: Overview and outlook, *Acta Autom. Sin.*, **46** (2020), 1539–1556. <https://doi.org/10.16383/j.aas.c200133>
22. I. Sekhi, S. Kovács, K. Nehéz, Enhancing decision-making in uncertain domains through optimized fuzzy logic systems, *Period. Polytech. Electr. Eng. Comput. Sci.*, **69** (2025), 63–78. <https://doi.org/10.3311/PPee.38729>
23. H. Jiang, Y. Tong, B. Song, C. Wang, J. Li, Q. Liu, et al., A robustness division based multi-population evolutionary algorithm for solving vehicle routing problems with uncertain demand, *Eng. Appl. Artif. Intell.*, **133** (2024), 108004. <https://doi.org/10.1016/j.engappai.2024.108004>

24. B. Rezaei, F. G. Guimaraes, R. Enayatifar, P. C. Haddow, Combining genetic local search into a multi-population imperialist competitive algorithm for the capacitated vehicle routing problem, *Appl. Soft Comput.*, **142** (2023), 110309. <https://doi.org/10.1016/j.asoc.2023.110309>
25. K. Q. Zhu, A diversity-controlling adaptive genetic algorithm for the vehicle routing problem with time windows, *ICTAI*, (2003), 176–183. <https://doi.org/10.1109/TAI.2003.1250187>
26. K. Deb, A. Pratap, S. Agarwal, T. Meyarivan, A fast and elitist multiobjective genetic algorithm: NSGA-II, *IEEE Trans. Evol. Comput.*, **6** (2002), 182–197. <https://doi.org/10.1109/4235.996017>
27. A. K. Ariyani, W. F. Mahmudy, Y. P. Anggodo, Hybrid genetic algorithms and simulated annealing for multi-trip vehicle routing problem with time windows, *Int. J. Electr. Comput. Eng.*, **8** (2018), 4713–4723. <https://doi.org/10.11591/ijece.v8i6.pp4713-4723>
28. S. Ropke, D. Pisinger, An adaptive large neighborhood search heuristic for the pickup and delivery problem with time windows, *Transp. Sci.*, **40** (2006), 393–546. <https://doi.org/10.1287/trsc.1050.0135>
29. L. Bertuccelli, H. L. Choi, P. Cho, J. How, Real-time multi-UAV task assignment in dynamic and uncertain environments, *AIAA GNC Conf.*, (2009). <https://doi.org/10.2514/6.2009-5776>
30. Z. Bi, X. Guo, J. Wang, S. Qin, G. Liu, Truck-drone delivery optimization based on multi-agent reinforcement learning, *Drones*, **8** (2024), 27. <https://doi.org/10.3390/drones8010027>



AIMS Press

© 2026 the Author(s), licensee AIMS Press. This is an open access article distributed under the terms of the Creative Commons Attribution License (<https://creativecommons.org/licenses/by/4.0>)