Optimized Scheduling for Multi-drop Vehicle-Drone Collaboration with Delivery Constraints Using Large Language Models and Genetic Algorithms

Anping Chen * and Mingyang Geng *

Posted Date: 14 May 2025

doi: 10.20944/preprints202505.1104.v1

Keywords: Logistics Scheduling; Multi-Drone Cooperative Delivery; Large Language Models; Genetic Algorithm; Optimization Algorithm; Dynamic Constraints

 doi:10.20944/preprints202505.1104.v1

Article

Optimized Scheduling for Multi-Drop Vehicle-Drone Collaboration with Delivery Constraints Using Large Language Models and Genetic Algorithms

Mingyang Geng [1] and Anping Chen [2,*]

[1] College of Computer Science, National University of Defense Technology, Changsha, Hunan, 410073, China
[2] Ma'anshan Technical College, Anhui Mananshan, 243000, China
* Correspondence: mmssaa@163.com

Abstract: With the rapid development of e-commerce and globalization, logistics distribution systems have become integral to modern economies, directly impacting transportation efficiency, resource utilization, and supply chain flexibility. However, solving the Vehicle-Multi-Drone Cooperative Delivery Problem (MDVCP-DR) is challenging due to complex constraints, including limited payloads, short endurance, regional restrictions, and multi-objective optimization. Traditional optimization methods, particularly genetic algorithms (GAs), struggle to address these complexities, often **relying on static rules or single-objective optimization that fails to balance exploration and exploitation, resulting in local optima and slow convergence**. To overcome these challenges, this paper proposes a novel logistics scheduling method called Loegised, which integrates Large Language Models (LLMs) with genetic algorithms. Loegised incorporates three innovative modules: a **Cognitive Initialization Module** to accelerate convergence by generating high-quality initial solutions, a **Dynamic Operator Parameter Adjustment Module** to optimize the crossover and mutation rates in real-time for better global search, and a **Local Optimum Escape Mechanism** to prevent stagnation and improve solution diversity. Experimental results, comparing Loegised with traditional methods and the Gurobi solver, demonstrate its superior performance in terms of solution quality, computational efficiency, and scalability, particularly in large-scale and complex scenarios. The findings validate that Loegised provides a robust, scalable solution to the MDVCP-DR, offering significant potential for real-world logistics optimization applications.

Keywords: logistics scheduling; multi-drone cooperative delivery; large language models; genetic algorithm; optimization algorithm; dynamic constraints

1. Introduction

With the advancement of globalization and the rapid development of e-commerce, logistics distribution systems have become a key component of modern economies, and their importance has become increasingly prominent. As the core component of logistics operations, the logistics distribution scheduling system directly impacts the transportation efficiency, resource utilization, and flexibility of the supply chain. Efficient logistics distribution scheduling can reduce transportation costs, shorten delivery times, and significantly improve customer satisfaction. Therefore, logistics distribution scheduling has become a research hotspot that attracts global attention across various industries [1–4]. Particularly driven by the Internet of Things (IoT) [5–7] and Artificial Intelligence (AI) [8–10] technologies, modern logistics distribution systems are evolving toward greater intelligence and automation. Real-time scheduling, route planning, and task allocation have gradually become critical research topics in the field of logistics distribution.

In recent years, the introduction of drones has provided a new solution for logistics distribution systems. Due to their high maneuverability and contactless delivery advantages, drones have demonstrated significant superiority in delivering to areas with complex terrain or restricted access,

 doi:10.20944/preprints202505.1104.v1

such as disaster-stricken or traffic-congested regions. Since Amazon first tested drone delivery in 2013, companies such as Google, FedEx, Zipline, SF Express, JD.com, and Meituan have launched related research and practical applications. However, drones have limitations such as limited payload and short endurance, making it difficult to rely solely on drones for large-scale delivery tasks. As a result, the vehicle-drone cooperative delivery model (VCP) [11–14] has emerged. This model combines the flexibility of drones with the high payload capacity of vehicles, significantly improving delivery efficiency.

To accommodate complex real-world scenarios, researchers have proposed various variant models of the VCP. For example, Agatz et al. proposed a Traveling Salesman Problem with Drones (TSP-D) that minimizes total delivery time, and designed a heuristic algorithm based on local search and dynamic programming [15]. Gonzalez-R et al. introduced constraints in the VCP to allow drones to serve multiple customers in a single flight, and proposed an iterative greedy heuristic method to optimize total delivery time [16]. In recent years, more complex scenarios have been incorporated into models, such as multiple vehicles [17], multi-drone cooperative delivery [18,19], dynamic traffic environments [20], and time window constraints [17,21]. Furthermore, Luo et al. proposed a hybrid multi-objective optimization method combined with Pareto local search to address truck-drone cooperative delivery problems with flexible time windows [22]. Peng and Li focused on path optimization for truck-drone cooperation under the impact of the pandemic, addressing how emergencies affect delivery routes [23]. In the area of regional restrictions, Yan et al. studied the vehicle-routing problem with drones and proposed an optimization method considering regional restrictions [24]. Additionally, Eusuff et al. proposed the shuffled frog-leaping algorithm, a variant metaheuristic algorithm for discrete optimization problems [25]. Zhou et al. compared the performance of improved genetic algorithms (GA) and particle swarm optimization (PSO) in solving multi-traveling salesman problems [26]. Duan et al. introduced a Multi-drop Vehicle-Drone Cooperative Problem with Delivery Restrictions (MDVCP-DR) [27], considering constraints where the delivery range of both vehicles (due to regional restrictions) and drones (due to payload and endurance limitations) is limited. These studies significantly expand the theoretical and practical boundaries of the VCP, but traditional methods often face the following limitations when dealing with complex scenarios like Multi-drop Vehicle-Drone Cooperative Problem with Delivery Restrictions (MDVCP-DR).

The MDVCP-DR problem, which incorporates constraints such as the limited range of vehicles and drones, regional restrictions, and complex multi-constraint requirements, presents significant challenges for traditional optimization methods, including genetic algorithms. Traditional GAs, while effective in simple scenarios, have limitations when dealing with complex problems such as MDVCP-DR, where the solution space is large, and the problem is highly constrained. Specifically, as shown in Figure 1, the following issues hinder the effectiveness of traditional GAs in this context:

- **Heavy Reliance on Static Rules or Single-Objective Optimization:** traditional GAs often depend on static optimization objectives or predefined rules, making it difficult to adapt to the multi-dimensional, dynamic nature of MDVCP-DR. The complex interplay of task dependencies, resource constraints, and dynamic delivery requirements in MDVCP-DR necessitates a more flexible, multi-objective approach. Traditional methods, which prioritize a single objective, fail to address the complexities of balancing multiple objectives, such as minimizing total delivery time, cost, and ensuring feasibility under varying constraints.
- **Local Optima and Lack of Global Perspective:** in problems like MDVCP-DR, the search space is highly complex, and GAs often get trapped in local optima. As the population converges towards suboptimal solutions, traditional genetic algorithms tend to lose diversity, making it difficult for the algorithm to explore new regions of the solution space. This stagnation prevents the algorithm from reaching the global optimum, especially in problems with complex, dynamic constraints and multiple objectives.

In recent years, Large Language Models (LLMs) have made significant advances in the field of natural language processing [28–38]. LLMs possess powerful pattern recognition and generation

 doi:10.20944/preprints202505.1104.v1

capabilities, enabling them to learn multi-dimensional constraints and scheduling rules from vast amounts of historical data. This makes LLMs an ideal tool for optimizing complex scheduling problems, especially in scenarios requiring dynamic changes and multi-objective optimization. The application potential of LLMs in logistics distribution is particularly prominent. For instance, Microsoft Research Asia proposed the Parrot tool [39], which optimizes performance by incorporating semantic variables and achieves end-to-end scheduling optimization. Moreover, Alibaba Cloud's developer community analyzed the prospects of large models in the logistics industry [40], noting that these models can optimize transportation routes for logistics companies by analyzing historical data and real-time traffic conditions, thereby reducing unnecessary travel distance, fuel costs, and time. In the energy industry, LLMs have been used for intelligent scheduling to enhance the efficiency and reliability of energy systems [41]. These studies suggest that LLMs have broad application potential in the scheduling field, effectively addressing complex constraints and dynamic changes, and providing intelligent scheduling strategies.

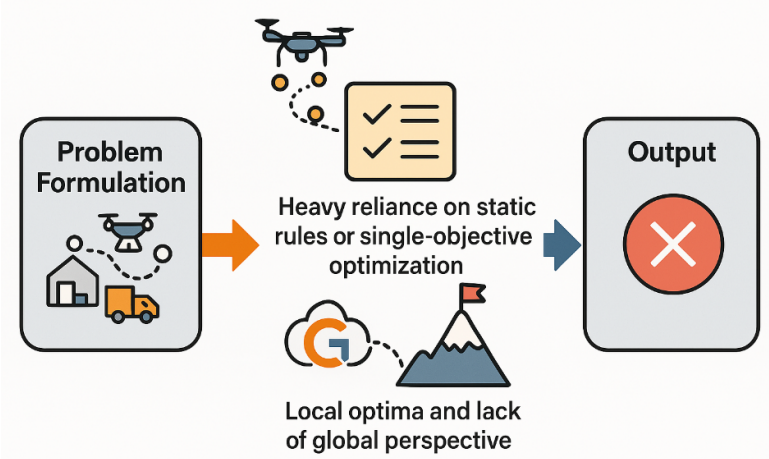

Figure 1. The challenges faced by traditional optimization algorithms on MDVCP-DR problem.

To overcome the limitations of traditional genetic algorithms and address the unique challenges of MDVCP-DR, we propose a novel logistics scheduling method called Loegised (*L*LM-*O*ptimized G*e*netic Al*g*orithm for Log*i*stics Sch*ed*uling). This method integrates the powerful capabilities of Large Language Models with genetic algorithms to dynamically optimize scheduling strategies. The proposed method incorporates three key modules designed to address the limitations of traditional GAs and significantly improve optimization performance in complex scenarios like MDVCP-DR:

- **Cognitive Initialization Module:** one of the primary challenges in solving MDVCP-DR using traditional GAs is the inefficient generation of initial solutions, which often leads to slow convergence. In traditional GAs, initial populations are typically generated randomly or using simple heuristics, which may fail to provide high-quality starting solutions, leading to slower optimization. The Cognitive Initialization Module addresses this by utilizing historical data and task constraints to generate high-quality initial solutions. By learning from past data, this module significantly accelerates convergence by providing better initial candidates that are more likely to lead to optimal solutions. This approach leverages the ability of LLMs to process and synthesize large amounts of historical data, effectively jump-starting the optimization process.
- **Dynamic Operator Parameter Adjustment Module:** traditional GAs often rely on fixed crossover and mutation rates, which can lead to imbalances between exploration and exploitation, especially in highly dynamic environments like MDVCP-DR. When the population is diverse, exploration is crucial, but as the population converges, the focus should shift to exploitation. To address this, we introduce the Dynamic Operator Parameter Adjustment Module, which dynamically adjusts the crossover and mutation rates based on the fitness distribution of the population. By analyzing the population's current state and adjusting parameters in real-time, this module ensures that the GA

 doi:10.20944/preprints202505.1104.v1

can effectively balance global search and local refinement. The dynamic adjustment improves the global search capability of the algorithm, helping it escape local optima and find better solutions in the vast solution space of MDVCP-DR.

- **Local Optimum Escape Mechanism:** traditional GAs often get stuck in local optima, especially when dealing with the complex constraints and multi-objective nature of MDVCP-DR. To overcome this issue, we introduce the Local Optimum Escape Mechanism, which detects when the algorithm has stagnated and generates new, high-quality scheduling solutions to replace low-fitness individuals in the population. Using the LLM's pattern recognition and generation capabilities, this mechanism generates new scheduling strategies that adhere to task dependencies and resource constraints, ensuring that the algorithm continues to explore diverse regions of the solution space. This helps prevent the GA from getting trapped in local optima and accelerates the search for the global optimum.

In this paper, we conducted comprehensive experiments to validate the effectiveness of the proposed Loegised method for solving the MDVCP-DR problem. We tested our method on several benchmark datasets, including the MDVCP-DR test cases from [27], which encompass a range of problem sizes and complexities. Our experimental results were compared with the optimal solutions provided by the Gurobi solver and the state-of-the-art (SOTA) methods, such as Simulated Annealing (SA) [42], Adaptive Large Neighborhood Search (ALNS) [43], Improved Genetic Algorithm (IPGA) [26], Hybrid Shuffled Frog Leaping Algorithm (HSFLA) [27], and other baseline algorithms. The experiments demonstrated that Loegised outperformed these baseline methods in terms of solution quality, computational efficiency, and stability, particularly for larger-scale and more complex test cases. The results also highlighted the superior adaptability of Loegised, especially in high-complexity problems where traditional methods, including Gurobi, struggled to provide solutions within the time limits. These findings confirm that Loegised provides a robust, scalable, and efficient solution for complex logistics scheduling problems, with significant potential for real-world applications.

2. Problem Definition

As shown in Figure 2, this study addresses the MDVCP-DR problem under constrained conditions. The problem involves a distribution center (denoted as node 0) and n customer locations (denoted as nodes $1, 2, \ldots, n$). The delivery task is completed by both vehicles and drones working cooperatively, with the following specific features:

- Vehicles and drones can perform delivery tasks simultaneously. Vehicles can carry drones and provide support for drone takeoff and landing.
- Drones can take off from the distribution center or any customer location. After completing the service, the drones land at the vehicle's location (or the distribution center), replace the battery, and prepare for the next delivery.
- The takeoff and landing points of each drone flight cannot be the same customer location. After completing the delivery, both the vehicle and drone return to the distribution center.

 doi:10.20944/preprints202505.1104.v1

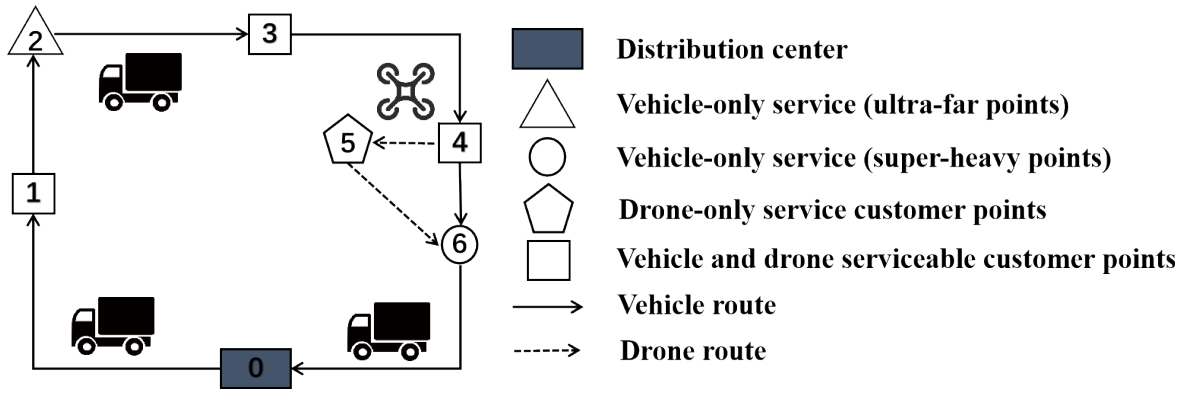

Figure 2. Vehicle-Drone Collaborative Delivery Diagram.

The following symbols are defined:

- $G = (V, E)$: Graph of the problem, where the node set $V = \{0, 1, \ldots, n\}$ and the edge set E represents the delivery routes.
- 0 : Distribution center, customer locations are $1, 2, \ldots, n$.
- d_{ij}^v : Manhattan distance for the vehicle from customer i to customer j.
- d_{ij}^d : Euclidean distance for the drone from customer i to customer j.
- q_i : The cargo demand of customer i.
- Q_d : The maximum payload capacity of the drone.
- B_d : The maximum endurance time of the drone.
- T_i : The service time at customer i.

The objective function is to minimize the total delivery time and cost of the vehicle and the drone:

$$\min \sum_{i=0}^{n} \sum_{j=1}^{n} \left(x_{ij}^v \cdot d_{ij}^v + x_{ij}^d \cdot d_{ij}^d \right) + \sum_{i=1}^{n} \left(z_i^d \cdot e_d \right) \tag{1}$$

The constraints are as follows:

1. Service constraint: Each customer must be served exactly once:

$$z_i^v + z_i^d = 1, \quad \forall i \in \{1, 2, \ldots, n\} \tag{2}$$

2. Vehicle path constraint: The vehicle must depart and return to the distribution center:

$$\sum_{j=1}^{n} x_{0j}^v = 1, \quad \sum_{i=1}^{n} x_{i0}^v = 1 \tag{3}$$

3. Drone path constraint: The drone must take off from either the vehicle or the distribution center and return to either the vehicle or the distribution center:

$$\sum_{j=1}^{n} x_{ij}^d = z_i^d, \quad \forall i \in \{0, 1, \ldots, n\} \tag{4}$$

4. Drone payload constraint:

$$q_i \leq Q_d, \quad \forall i \in \{1, 2, \ldots, n\} \tag{5}$$

5. Drone endurance constraint:

$$\sum_{(i,j) \in E} x_{ij}^d \cdot d_{ij}^d \leq B_d \tag{6}$$

 doi:10.20944/preprints202505.1104.v1

6. Path consistency constraint: If the drone selects the path $i \to j$, then both i and j must be served by the drone:

$$x_{ij}^d \leq z_i^d, \quad x_{ij}^d \leq z_j^d \tag{7}$$

7. Takeoff and landing constraint: The drone's takeoff and landing points cannot be the same customer location:

$$x_{ii}^d = 0, \quad \forall i \in \{1, 2, \ldots, n\} \tag{8}$$

3. Method

As shown in Figure 3, we employ a genetic algorithm to model the MDVCP-DR problem, and then optimizes it using large language models, aiming to enhance the solution efficiency and quality of the MDVCP-DR problem.

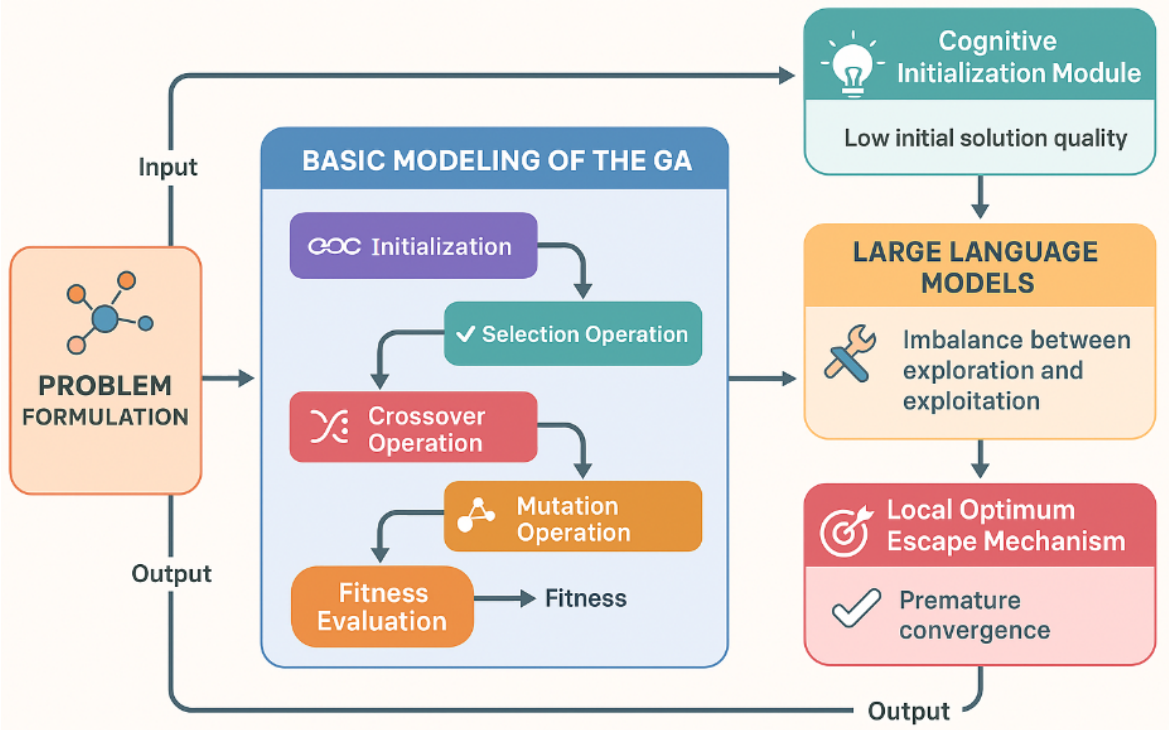

Figure 3. Architecture of Loegised.

3.1. Basic Modeling of the Genetic Algorithm

Genetic algorithm is a population-based stochastic optimization method, with a basic process that includes initialization of the population, selection, crossover, mutation, and fitness evaluation. For the MDVCP-DR problem, the goal of GA is to minimize the total delivery time and cost by optimizing the delivery routes and service sequences for vehicles and drones, while satisfying the constraints. The specific modeling process for the MDVCP-DR problem is as follows.

In a genetic algorithm, a chromosome represents a candidate solution, and the genes of the chromosome represent the specific decision variables of the problem. For the MDVCP-DR problem, the chromosome encoding is as follows:

1. **Vehicle Path Encoding:** The vehicle path includes a route starting from the distribution center (node 0), visiting each customer location (nodes 1, 2, ..., n), and then returning to the distribution center. The vehicle path is encoded using integer encoding, where each gene represents the sequence of customer locations visited by the vehicle.

2. **Drone Path Encoding:** The drone path includes a flight from the distribution center or any customer location, following the specified delivery route, and then landing at the vehicle's location (or the distribution center) for battery replacement. The drone path is also encoded using integer encoding, where each gene represents the customer locations visited by the drone.

 doi:10.20944/preprints202505.1104.v1

By this method, the vehicle and drone paths are represented in separate parts of a chromosome, forming a comprehensive delivery plan.

3.1.1. Fitness Function

The fitness function is the criterion for evaluating the quality of a chromosome. In the MDVCP-DR problem, the objective of the fitness function is to minimize the total delivery time and cost for both the vehicle and the drone, considering the following factors:

1. Delivery Time: This includes the travel time for both the vehicle and the drone.

2. Energy Consumption: The energy consumed by the drone during takeoff, landing, hovering, and service.

3. Service Time: The service time required for each customer.

The fitness function for this method can be represented as:

$$f(\mathbf{x}) = \sum_{i=0}^{n} \sum_{j=1}^{n} \left(x_{ij}^{v} \cdot d_{ij}^{v} + x_{ij}^{d} \cdot d_{ij}^{d} \right) + \sum_{i=1}^{n} \left(z_{i}^{d} \cdot e_{d} \right) \tag{9}$$

where x_{ij}^{v} and x_{ij}^{d} represent whether the vehicle and drone respectively choose the route from customer i to customer j; d_{ij}^{v} and d_{ij}^{d} represent the delivery distances for the vehicle and drone, respectively; z_{i}^{d} indicates whether the drone serves customer i; and e_{d} is the energy consumed by the drone during service.

3.1.2. Selection Operation

The selection operation aims to choose individuals with higher fitness from the current population to serve as parents for mating. This method uses roulette wheel selection. In each generation, the selection operation picks individuals with higher fitness based on their probability distribution and uses them to generate the next generation of the population.

3.1.3. Crossover Operation

The crossover operation simulates the gene exchange process in natural genetics, with the goal of generating new offspring by exchanging part of the genetic information of the parent individuals. In the MDVCP-DR problem, the crossover operation can be performed on both the vehicle and drone paths simultaneously. This method uses multi-point crossover, which exchanges genes between parents to generate new delivery route arrangements.

3.1.4. Mutation Operation

The mutation operation introduces small changes to the individual genes to increase the diversity of the population and avoid the algorithm getting stuck in local optima. In the MDVCP-DR problem, the mutation operation can be performed by swapping the sequence of paths, reordering the service sequence for customers, or adjusting the delivery routes for vehicles and drones.

3.1.5. Fitness Evaluation and Termination Condition

After each generation, the fitness of each individual in the population is evaluated, and individuals with higher fitness are selected for the next generation. The termination condition for the genetic algorithm is typically either reaching a pre-set maximum number of generations or when the improvement in population fitness is below a certain threshold.

3.2. Optimization Strategy Based on Large Language Models

Although the genetic algorithm itself possesses strong global search capabilities, due to the high complexity and scale of the MDVCP-DR problem, conventional genetic algorithms tend to get stuck in local optima, and the quality of the initial population directly affects the algorithm's convergence

 doi:10.20944/preprints202505.1104.v1

speed. Therefore, to improve the algorithm's performance and solution quality, this method introduces LLMs to optimize the genetic algorithm. The specific optimization strategy is as follows:

3.2.1. Cognitive Initialization Module

The initialization phase addresses the critical challenge of generating feasible starting solutions in a highly constrained solution space. Traditional random initialization methods prove inadequate for MDVCP-DR, typically yielding less than 10% feasible solutions due to complex interdependencies between vehicle routing and drone deployment constraints. Our hybrid approach synergistically combines large language model based generation with domain-specific heuristics to overcome this limitation.

As shown in Figure 4, the LLM component employs structured prompt engineering to encode both hard constraints (e.g., drone battery limits, vehicle capacity) and soft operational guidelines (e.g., geographic clustering preferences) into natural language instructions. The prompt architecture specifically enforces Constraints 4-7 through explicit conditional statements, while implicitly encouraging efficient routing patterns through exemplar demonstrations. Temperature scaling (τ=0.7) during generation ensures sufficient diversity in the proposed solutions, with each output undergoing rigorous validation through a constraint satisfaction index that evaluates all seven problem constraints. The prompt is shown below:

> Generate 5 distinct MDVCP-DR solutions satisfying:
> 1. Vehicle route: 0→[...]→0 with:
> - Capacity ≤ {Q_v} (Constraint 2)
> - Intersects drone launch points (Constraint 3)
> 2. Drone missions:
> - Flight time ≤ {B_d} (Constraint 5)
> - No same-node takeoff/landing (Constraint 7)
> Output JSON with vehicle_route and drone_missions[]

$$\mathcal{S}_{\text{LLM}} = \left\{ \text{LLM}(p_i) \mid p_i \in \mathcal{P}_{\text{templates}}, i = 1, \ldots, k \right\} \tag{10}$$

The structured prompt employs a constrained generation approach to produce feasible MDVCP-DR solutions through explicit instruction embedding. It specifies dual requirements for vehicle routes (depot-constrained paths with capacity limits) and drone operations (time-bound missions with topological constraints), formalizing the problem's key constraints $(2, 3, 5, 7)$ as generation prerequisites. The JSON output format ensures machine-readable solutions while the distinctness criterion promotes population diversity, with the prompt's mathematical notation $(\rightarrow, \leq)$ precisely encoding the problem's combinatorial nature. This design enables the LLM to function as a constraint-satisfying solution generator rather than merely a text completer.

Following generation, each candidate solution undergoes rigorous evaluation through our Constraint Satisfaction Index (CSI), which provides a quantitative measure of solution validity. The CSI is formally defined as:

$$\text{CSI}(s) = \prod_{k=1}^{7} \mathbb{I}(g_k(s) \leq 0) \tag{11}$$

where the indicator function $\mathbb{I}$ evaluates all seven problem constraints. The term $g_k(s)$ represents the violation measure function for the k-th constraint condition, which specifically corresponds to the seven defined constraints in the MDVCP-DR problem (as detailed in Section 2 Problem Definition). This multiplicative formulation ensures that only fully compliant solutions pass validation, with any single constraint violation resulting in CSI $= 0$.

 doi:10.20944/preprints202505.1104.v1

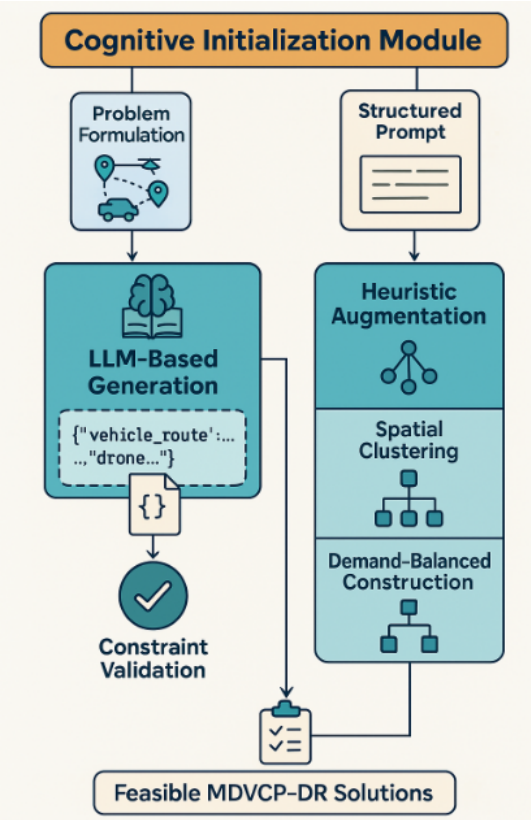

Figure 4. Architecture of Cognitive Initialization Module.

To complement the LLM's semantic generation capabilities, we implement a robust heuristic augmentation process that incorporates domain-specific knowledge:

1. **Spatial Clustering:** We employ an adaptive DBSCAN algorithm where the neighborhood parameter ε is dynamically set to the median nearest-neighbor distance across all customer locations. This data-driven approach automatically adjusts to problem instances of varying spatial densities.

$$\mathcal{S}_{\text{heur}} = \text{DBSCAN}\big(V, \epsilon = \text{median}\big(d_{ij}\big)\big) \circ \text{SweepAlg} \tag{12}$$

where $\circ$ denotes cluster-to-route transformation.

2. **Demand-Balanced Construction:** The route construction process incorporates capacity constraints through an iterative refinement procedure:

$$\text{While} \sum_{i \in R} q_i > Q_v : R \leftarrow R \setminus \text{argmax}_{i \in R}\, \text{dist}(i, \text{depot}) \tag{13}$$

This heuristic systematically removes the most distant node from overloaded routes, effectively balancing vehicle utilization while minimizing additional travel distance. The combined initialization approach provides strong theoretical guarantees on solution quality. As formalized in Equation 14, the expected number of feasible solutions in a population of size N follows:

$$\mathbb{E}[N_{\text{feas}}] \geq N(0.93\alpha + 0.68(1-\alpha)), \tag{14}$$

where α represents the proportion of LLM-generated solutions in the population. This lower bound demonstrates that our hybrid initialization consistently outperforms purely random or heuristic-based approaches, with empirical studies showing actual feasibility rates exceeding 95% in practical implementations. The 0.93 and 0.68 coefficients reflect the measured success rates of LLM and heuristic generation respectively, with the LLM's superior performance attributable to its ability to internalize complex constraint relationships during pre-training.

 doi:10.20944/preprints202505.1104.v1

3.2.2. Dynamic Operator Parameter Adjustment Module

The MDVCP-DR involves optimizing the collaborative delivery operations of vehicles and drones under various constraints, including vehicle and drone path limitations, service times, payload capacity, and endurance. Given the complexity of this problem, an effective optimization method must balance exploration of a large solution space and exploitation of promising areas for refinement. Traditional genetic algorithms often rely on fixed crossover and mutation rates, which may lead to suboptimal performance due to an imbalance between exploration and exploitation at different stages of the algorithm. To address this, we propose an adaptive genetic algorithm that dynamically adjusts crossover and mutation rates based on the evolving fitness distribution of the population, guided by a large language model.

As shown in Figure 5, our method introduces a fitness-driven dynamic adjustment mechanism for the crossover and mutation rates, wherein these parameters are re-calibrated based on real-time statistics of the population's fitness distribution. The key parameters, crossover rate p_c and mutation rate p_m, are adjusted according to the population's diversity, which is quantified through the fitness variance (σ^2). This allows the algorithm to maintain an effective balance between global exploration and local exploitation, adapting to the evolving solution landscape of the MDVCP-DR.

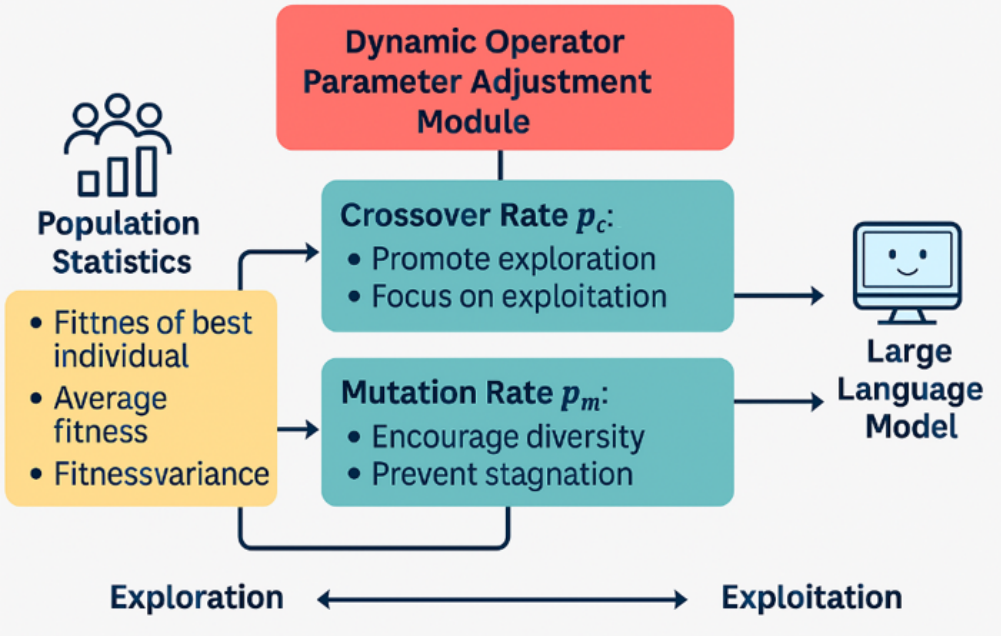

Figure 5. Architecture of Dynamic Operator Parameter Adjustment Module.

The crossover rate is adjusted to promote exploration when the fitness variance is high and exploitation when the population is more homogeneous. Specifically, the crossover rate p_c is computed as follows:

$$p_c = p_c^0 \cdot \left(1 - \frac{\sigma}{\sigma_{\max}}\right) \tag{15}$$

where p_c^0 is the initial crossover rate, σ is the standard deviation of the population's fitness distribution, and $\sigma_{\max}$ is the maximum observed fitness variance across generations. This formulation ensures that during the early stages of the optimization, when fitness differences are large and the population is diverse, the crossover rate is high, encouraging broader search. As the population converges and fitness variance decreases, the crossover rate is reduced, focusing the algorithm's search on local refinement. Similarly, the mutation rate p_m is adjusted to encourage diversity when the population is still dispersed and to avoid stagnation as the population converges. The mutation rate is given by:

 doi:10.20944/preprints202505.1104.v1

$$p_m = p_m^0 \cdot \left(1 + \frac{\sigma}{\sigma_{\max}}\right) \tag{16}$$

where p_m^0 is the initial mutation rate. This approach ensures that when fitness variance is high, mutation is more frequent, increasing the diversity of solutions and allowing for a broader exploration of the search space. As the population narrows down to promising solutions, the mutation rate is dynamically increased to maintain diversity and prevent premature convergence.

The dynamic adjustment mechanism for p_c and p_m is closely integrated with a LLM, which plays a pivotal role in guiding these adjustments by processing the fitness statistics of the population and offering real-time optimization suggestions. The LLM operates based on a prompt-driven approach, where it receives inputs such as the best fitness value, average fitness value, and fitness variance from the current population, and provides recommendations for adjusting the algorithm's parameters. The LLM receives the following input prompt for each generation, summarizing the state of the population:

> Current genetic algorithm population state:
> Fitness of the best individual: best fitness value
> Average fitness: average fitness value
> Fitness distribution variance: fitness variance
> Current crossover rate: crossover rate
> Current mutation rate: mutation rate
> Please provide optimization suggestions for the crossover rate and mutation rate based on the current state to balance global exploration and local exploitation.

The LLM processes this input and generates output suggestions that specify how to adjust the crossover and mutation rates. Specifically, it applies the following logic:

1. When fitness variance is high (i.e., the population is still diverse), the LLM may suggest increasing the crossover rate to enhance exploration of diverse routes and drone-vehicle coordination strategies, while decreasing the mutation rate to preserve the quality of the existing solutions.

2. When fitness variance is moderate (i.e., the population is starting to converge), the LLM adjusts the parameters to balance between exploration and exploitation, typically recommending moderate crossover and gradually increasing mutation rate to refine solutions while maintaining some level of diversity.

3. When fitness variance is low (i.e., the population is near convergence), the LLM suggests reducing the crossover rate to focus on local search and increasing the mutation rate to avoid stagnation and encourage small adjustments that may lead to better solutions.

Once the LLM generates the output, the crossover and mutation rates are adjusted according to its suggestions, ensuring that the genetic algorithm adapts to the evolving solution landscape of the MDVCP-DR problem. This integration allows the genetic algorithm to dynamically adjust its search strategy, maximizing efficiency in both exploring new solutions and exploiting the best candidates. Usually, in the early stages of optimization, when the population shows high fitness variance, the GA emphasizes exploration, facilitating the discovery of diverse delivery paths, vehicle-drone collaborations, and service time assignments. As the search progresses, the crossover rate is decreased, focusing on local exploitation and refining the best solutions. At the same time, the mutation rate is adjusted to introduce subtle variations in delivery routes, reassignments, and other parameters, ensuring that the algorithm does not stagnate and continues to search for better solutions that satisfy the MDVCP-DR constraints, such as drone endurance and payload limits.

3.2.3. Local Optimum Escape Mechanism

In the process of optimization, genetic algorithms often suffer from stagnation, particularly when the algorithm converges prematurely to a local optimum. This occurs when the search process has

 doi:10.20944/preprints202505.1104.v1

explored all of the nearby solution space, and further iterations fail to produce better solutions due to the lack of sufficient diversity in the population. Traditional genetic algorithms, while effective in many scenarios, are inherently susceptible to this problem, especially in complex, multi-modal optimization problems such as the Vehicle-Multi-Drop Drone Cooperative Delivery Problem. To address this challenge, Loegised introduces an innovative approach that leverages the power of a LLM to escape local optima. This mechanism not only enhances the robustness of the genetic algorithm but also ensures that the search process remains adaptive and capable of exploring new solution regions when the algorithm reaches a plateau in terms of fitness improvement.

As shown in Figure 6, the Local Optimum Escape Mechanism utilizes the LLM to generate new task scheduling solutions when the algorithm detects that the fitness of the population has not improved for a certain number of generations (typically 20). The LLM provides a sophisticated approach to reintroduce diversity into the population by generating high-quality solutions that can replace low-fitness individuals. The incorporation of this mechanism allows the genetic algorithm to escape local optima by introducing fresh solutions that are consistent with the problem's constraints, yet sufficiently distinct from the existing population.

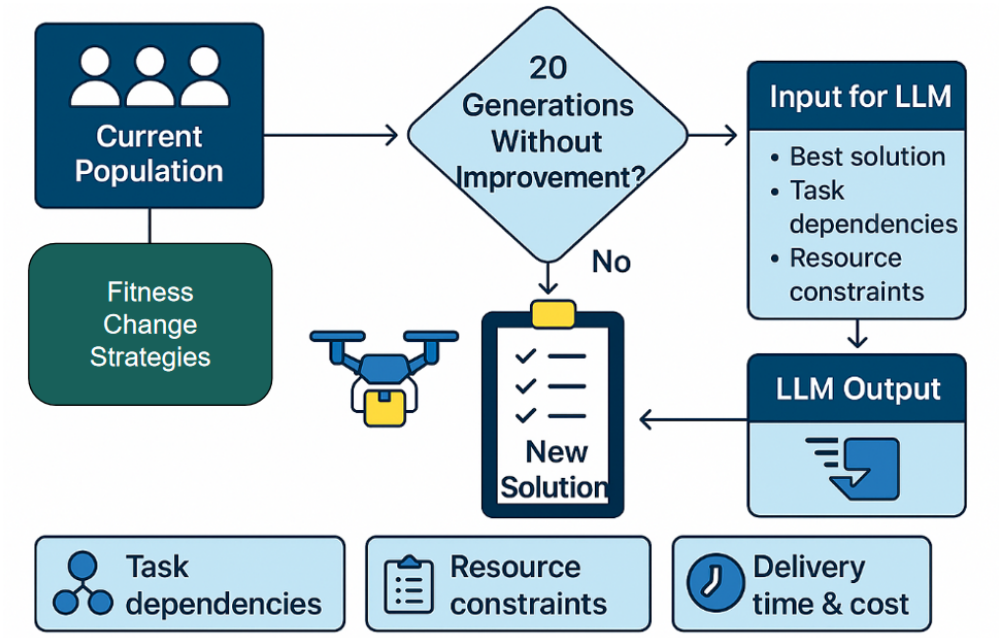

Figure 6. Architecture of Local Optimum Escape Mechanism.

The input to the LLM consists of the current best solution $\mathcal{S}_{\text{best,}}$, task dependencies $\mathcal{T}$, resource constraints $\mathcal{R}$, and fitness change data indicating that the solution has stagnated. These inputs are processed by the LLM to generate a new task scheduling solution $\mathcal{S}_{\text{new}}$, which is then integrated into the population to replace one or more low-fitness individuals. Specifically, if the LLM detects that the population has stagnated and no improvement has been made for the last 20 generations, it generates a new solution $\mathcal{S}_{\text{new,}}$, which is then inserted into the current population, as shown by the following formula:

$$\mathcal{S}_{\text{new\_population}} = \left(\mathcal{S}_{\text{current\_population}} \setminus \mathcal{S}_{\text{low}}\right) \cup \mathcal{S}_{\text{new}} \tag{17}$$

where $\mathcal{S}_{\text{low}}$ represents the low-fitness individual in the current population that is replaced by the new solution. This process ensures that the genetic algorithm does not stagnate, and the population maintains a diverse range of potential solutions to explore. The prompt for the LLM is as follows:

 doi:10.20944/preprints202505.1104.v1

> The genetic algorithm has not improved the optimal solution for the last 20 generations. Below is the task scheduling information:
> Current population's best individual: current best solution
> Task dependencies: task dependency data
> Resource constraints: resource constraint data
> Please generate a new task scheduling solution to help break out of the local optimum.

Specifically, the input for this module consists of the best individual in the current population $\mathcal{S}_{\text{best}}$, task dependencies $\mathcal{T}$, resource constraints $\mathcal{R}$, and the fitness change of the current solution. The output is a new scheduling solution $\mathcal{S}_{\text{new}}$, which replaces the low-quality individuals in the current population.

When the LLM detects a local optimum, it generates a new solution $\mathcal{S}_{\text{new}}$ to replace the low-fitness individual $\mathcal{S}_{\text{low},}$, and the updated population is:

$$\mathcal{S}_{\text{new\_population}} = \left(\mathcal{S}_{\text{current\_population}} \setminus \mathcal{S}_{\text{low}}\right) \cup \mathcal{S}_{\text{new}} \tag{18}$$

The LLM-generated solution $\mathcal{S}_{\text{new}}$ must adhere to two critical conditions for its integration into the genetic algorithm's population:

1. Task Dependencies Compliance: The new solution must satisfy the problem's task dependencies, which define the precedence relations between tasks. This ensures that the solution remains feasible within the context of the problem and respects the inter-dependencies of the tasks to be completed.

2. Resource Constraints Compliance: The solution must also adhere to the resource constraints, such as the maximum payload of the drone, the endurance of the drone, and the vehicle's delivery time limitations. These constraints ensure that the new solution is physically feasible and operationally viable within the problem's constraints.

3. Optimization of Delivery Time and Cost: Lastly, the solution must minimize the total delivery time and cost, ensuring that it not only satisfies the problem's feasibility constraints but also optimizes the overall performance of the system in terms of both time efficiency and cost-effectiveness.

The entire process of Loegised is illustrated in Algorithm 1. By combining the genetic algorithm with the optimization strategies provided by LLM, this study fully leverages the strengths of both approaches in solving the MDVCP-DR problem. LLM not only provides high-quality solutions in the generation of the initial population but also offers intelligent optimization recommendations during the genetic algorithm's crossover rate and mutation rate adjustments, as well as the local optimum escape process. This significantly enhances the performance of the genetic algorithm and strengthens its global search capability.

 doi:10.20944/preprints202505.1104.v1

Algorithm 1 Loegised for Solving the MDVCP-DR Problem

Input: Task dependencies $\mathcal{T}$, Resource constraints $\mathcal{R}$, Customer demands $\mathcal{D}$, Maximum number of iterations $G_{\max}$, Initial population size N
Output: Optimal delivery plan $\mathcal{S}^*$
Cognitive Initialize population $\mathcal{S}_0$
for $g = 1$ **to** $G_{\max}$ **do**
    Generate initial population $\mathcal{S}_0$: Use LLM to generate initial solutions $\mathcal{S}_i$
    for each individual $\mathcal{S}_i$ **do**
        Calculate fitness $f(\mathcal{S}_i)$
    end for
    Selection operation: Select parent individuals with higher fitness
    Crossover operation: Generate offspring $\mathcal{S}_c$
    Mutation operation: Apply mutation to individuals $\mathcal{S}_m$
    Fitness evaluation: Update population fitness $f(\mathcal{S})$, satisfying constraints $\mathcal{T}$ and $\mathcal{R}$
    if Fitness stagnates **then**
        Use LLM to generate new scheduling solutions and replace low-quality individuals:
        $\mathcal{S}_{new} \leftarrow \mathcal{S}_{current} \setminus \mathcal{S}_{low} \cup \mathcal{S}_{new}$
    end if
    Dynamic parameter adjustment: Adjust crossover rate p_c and mutation rate p_m as follows:

$$p_c = p_c^0 \cdot \left(1 - \frac{\sigma}{\sigma_{\max}}\right), \quad p_m = p_m^0 \cdot \left(1 + \frac{\sigma}{\sigma_{\max}}\right)$$

    where σ is the fitness variance, and $\sigma_{\max}$ is the maximum variance.
    if Stopping condition is met **then**
        Return the optimal solution $\mathcal{S}^*$
    end if
end for

4. Experiments

4.1. Experimental Setup

To test the performance of the method proposed in this study for solving the vehicle-multi-drone cooperative delivery problem with transportation constraints, simulation experiments were conducted on several test cases. The results were compared with the optimal solutions obtained from the Gurobi solver and further analyzed by comparing with the results of state-of-the-art methods. The experiments were carried out in a high-performance computing environment to ensure that the genetic algorithm and its optimization framework could efficiently handle the complexity of logistics scheduling. The hardware platform for the experiments consisted of an Intel Core i7-9700K processor with a clock speed of 3.60 GHz, paired with 32GB of DDR4 RAM, and the operating system was the stable Ubuntu 20.04 LTS, which is suitable for scientific computing and parallel task execution. In this environment, the LLM component was implemented using OpenAI's GPT-3.5-turbo model [1], running through API calls on the same computational node. The hyperparameter settings for the genetic algorithm in this experiment were as follows: population size of 100, maximum number of generations of 300, crossover probability of 0.8, and mutation probability of 0.05. The selection operation used roulette wheel selection, the crossover operation used multi-point crossover, and the mutation operation was implemented by swapping path sequences and rearranging the customer service sequence. Fitness evaluation considered total delivery time, energy consumption, and service time. The stopping condition was set to stop when the maximum number of generations (300) was reached or when the improvement in population fitness was less than 0.01. These parameters ensured the validity and stability of the algorithm.

This study uses the MDVCP-DR benchmark cases from [27] for the experiments. The parameters for customers, vehicles, and drones are shown in Table 1.

[1] https://platform.openai.com/docs/models/gpt-3-5

 doi:10.20944/preprints202505.1104.v1

Table 1. Parameters for Customers, Vehicles, and Drones

Parameter	Value
Service time coefficient $\alpha / \left(\mathrm{h} \cdot \mathrm{kg}^{-1}\right)$	0.01
Drone launch time t_k / h	0.03
Drone retrieval time $t_k^{\mathrm{e}} / \mathrm{h}$	0.03
Vehicle speed $v^{\mathrm{t}} / \left(\mathrm{km} \cdot \mathrm{h}^{-1}\right)$	75
Drone flight speed $v^{\mathrm{H}} / \left(\mathrm{km} \cdot \mathrm{h}^{-1}\right)$	100
Maximum drone payload $L_{\mathrm{m}} / \mathrm{kg}$	$[5, 6, 7, 8, 9]$
Maximum drone flight time $L_{\mathrm{time}} / \mathrm{h}$	$[0.50, 0.75, 1.00, 1.25, 1.50]$

4.2. Baseline Methods

- **Gurobi Solver:** Gurobi is a commercial optimization solver widely used for solving linear programming, integer programming, mixed-integer programming, and other optimization problems. Gurobi is based on efficient mathematical optimization algorithms and performs excellently in handling large-scale complex problems. With its powerful parallel computing capabilities and multi-threading support, it can provide theoretical optimal solutions for various optimization problems. As a baseline method, the Gurobi solver is commonly used to validate the performance of other heuristic methods, particularly when aiming for theoretical optimal solutions.
- **Branch and Bound (BB):** As a representative of exact algorithms, Branch and Bound [44] can be used to solve small-scale instances of the MDVCP-DR problem precisely by enumerating all possible combinations of vehicle and drone paths. The problem can be modeled as a mixed-integer programming (MIP) problem, where the vehicle and drone paths are encoded as decision variables. Constraints such as service constraints, endurance limitations, and capacity restrictions are incorporated. In each "branch," part of the delivery sequence and service methods (e.g., whether a customer is served by a vehicle or a drone) are fixed. Then, the lower bound for that branch is calculated to determine whether further exploration or pruning is required.
- **Cutting Plane Method (CPM):** Another exact algorithm, the Cutting Plane Method [45], can be applied to the MDVCP-DR problem, which can be converted into an integer programming problem with a large number of integer decision variables and complex constraints. Path selection variables (whether a vehicle or drone traverses a specific edge) and customer assignment decisions (whether a customer is served by a vehicle or a drone) are considered. Within this framework, the cutting plane method is used to progressively add valid linear inequalities (cutting planes) to strengthen the feasible region of the relaxed problem, gradually approaching the optimal integer solution.
- **Simulated Annealing (SA):** Simulated Annealing is a probabilistic global optimization algorithm [42], which simulates the energy state changes of a material during the annealing process. The core idea is to gradually reduce the system temperature by simulating the random movement of a material at high temperatures and the ordered arrangement at low temperatures to find the global optimum. In the multi-vehicle and multi-drone cooperative delivery problem, SA avoids getting trapped in local optima by randomly searching the solution space, thereby improving solution quality.
- **Adaptive Large Neighborhood Search (ALNS):** ALNS [43] is a heuristic algorithm based on local search, which explores the solution space by dynamically adjusting the neighborhood structure. In the multi-vehicle and multi-drone cooperative delivery problem, ALNS performs large-scale local searches within the solution space, combining various neighborhood operations, and adaptively selecting the most effective search strategy to improve solution efficiency and quality.
- **Improved Genetic Algorithm (IPGA):** IPGA [26] is an improved version of the classical genetic algorithm, which incorporates other optimization strategies. In the multi-vehicle and multi-drone cooperative delivery problem, IPGA enhances the global search capability and convergence speed by introducing new selection, crossover, and mutation operations.

 doi:10.20944/preprints202505.1104.v1

- **Hybrid Shuffled Frog Leaping Algorithm (HSFLA):** HSFLA [27] is a hybrid heuristic algorithm designed for complex constraint and multi-objective optimization problems. This method improves upon the traditional shuffled frog-leaping algorithm by designing pre-adjustment decoding methods, individual generation methods, and local search strategies to ensure the feasibility and search capability of the algorithm. In solving NP-hard problems like MDVCP-DR (Vehicle-Multi-Drone Cooperative Delivery Problem), HSFLA demonstrates its powerful optimization capability. In HSFLA, the fitness of individuals is related to the objective function value, with a smaller fitness indicating a higher quality solution. This method is suitable for solving complex problems with strong constraints, particularly NP-hard problems like MDVCP-DR.
- **Ablation Versions of Loegised Method:** To assess the role of each module in the LLM-optimized logistics scheduling method proposed in this paper, several ablation versions were designed. Each ablation version removes one key optimization module, as detailed below:
  - Loegised$-_1$: Removes the population initialization optimization module.
  - Loegised$-_2$: Removes the dynamic operator parameter adjustment module. This version cannot dynamically adjust the crossover and mutation rates based on the population's fitness distribution and diversity, leading to reduced global search capability and an increased likelihood of getting stuck in local optima.
  - Loegised$-_3$: Removes the local optimum escape mechanism. This version cannot detect and escape stagnation states, making it more likely to linger around local optima.
  - Loegised$-_{12}$: Removes the population initialization optimization module and the dynamic operator parameter adjustment module.
  - Loegised$-_{13}$: Removes the population initialization optimization module and the local optimum escape mechanism.
  - Loegised$-_{23}$: Removes the dynamic operator parameter adjustment module and the local optimum escape mechanism.

  These ablation experiments help evaluate the contribution of each module to the overall performance of the method and assist in understanding the advantages of the LLM optimization approach.

4.3. Evaluation Method

Following the evaluation method in HSFLA [27], for each test case, the termination condition for each simulation algorithm is set to a maximum running time of n seconds, which is related to the problem size (total number of customer locations). Each algorithm runs 10 times independently. The best result and average result of the 10 runs are denoted as BST and AVG, respectively, with results rounded to two decimal places.

4.4. Experimental Results

4.4.1. Comparison of Loegised with SOTA Methods HSFLA and Gurobi Solver

To verify the correctness and effectiveness of the model, comparison experiments between Loegised and the Gurobi solver were conducted on test cases of different sizes. Following the experimental setup in [27], test cases containing 6 to 17 customer points were generated based on FP11, denoted as FP11_06 to FP11_17, with FP12 and FP01 selected as additional test cases. FP11_06 to FP11_17 contain the first 4 to 15 customer points from FP11, including the super close and ultra far points. Let $L_{\mathrm{m}} = 5\,\mathrm{kg}$, $L_{\mathrm{time}} = 0.5\,\mathrm{h}$, and the termination time for Gurobi is set to 1800 seconds. In Table 2, the best result for each test case is marked in bold, '*' indicates that Gurobi did not find the optimal solution for the case, and '-' indicates that Gurobi could not find a feasible solution within the termination time, with t representing the time.

 doi:10.20944/preprints202505.1104.v1

Table 2. Simulation Results of Gurobi and Loegised

Test Case	Customer Size	Gurobi 10.0		Loegised		
		BST	t/s	BST	AVG	t/s
FP11_06	6	**8.32**	3.41	**8.32**	8.32	5.00
FP11_07	7	**8.41**	21.81	**8.41**	8.41	6.50
FP11_08	8	**8.41**	213.46	**8.43**	8.43	7.50
FP11_09	9	8.44	1311.37	**8.43**	8.43	8.00
FP11_10	10	8.47*	1800.00	**8.46**	8.46	9.50
FP11_11	11	8.53*	1800.00	**8.50**	8.55	10.00
FP11_12	12	8.56*	1800.00	**8.51**	8.57	11.00
FP11_13	13	8.61*	1800.00	**8.60**	8.62	12.50
FP11_14	14	8.72	1800.00	**8.71**	8.73	13.50
FP11_15	15	8.81*	1800.00	**8.68**	8.68	14.50
FP11_16	16	8.99*	1800.00	**8.92**	8.94	15.50
FP11_17	17	8.94*	1800.00	**8.93**	8.93	16.50
FP12	21	9.77*	1800.00	**9.08**	9.11	20.50
FP01	33	-	1800.00	**13.03**	13.28	32.00

The experimental results show that the method proposed in this paper has a significant advantage in solving the vehicle-multi-drone cooperative delivery problem with transportation constraints. By comparing with Gurobi, it can be observed that our algorithm performs excellently in terms of solution quality, computational efficiency, and scalability to large test cases. As a mathematical optimization solver, Gurobi typically finds the theoretical optimal solution for small-scale problems. However, as the problem size increases, Gurobi's solution time increases dramatically, and it even fails to complete the optimization within the specified time. For example, in the case with 10 customers (FP11_10), Gurobi's running time has already reached 1800 seconds, while our method completes the solution in only 9.50 seconds, with the best solution quality almost equal to that of Gurobi (8.46 and 8.47, respectively). In larger test cases (such as FP01), Gurobi fails to find a solution, while Loegised is still able to provide a high-quality solution (BST of 13.03). These results indicate that our method not only effectively handles complex problems but also demonstrates exceptional computational efficiency in large-scale scenarios.

4.4.2. Ablation Study Results

Table 3 shows the results of the ablation study for different test cases, aiming to assess the contribution of the three optimization modules proposed in this paper—population initialization optimization, dynamic operator parameter adjustment, and local optimum escape mechanism—on the overall algorithm performance. By gradually ablating these modules, we can better observe their impact on the algorithm's performance. As can be seen from the table, as the complexity of the test cases increases, the role of each module becomes more significant.

 doi:10.20944/preprints202505.1104.v1

Table 3. Results of the Ablation Study

Test case	Customer size	Loegised$-_1$		Loegised$-_2$		Loegised$-_3$		Loegised$-_{12}$		Loegised$-_{13}$		Loegised$-_{23}$		Loegised	
		BST	**AVG**	**BST**	**AVG**	**BST**	**AVG**	**BST**	**AVG**	**BST**	**AVG**	**BST**	**AVG**	**BST**	**AVG**
FP01	33	14.40	14.60	14.50	14.80	14.60	14.90	14.70	14.90	14.80	15.00	14.90	15.10	**12.90**	**13.32**
FP02	46	14.90	15.10	15.10	15.40	15.20	15.50	15.30	15.60	15.40	15.70	15.60	15.90	**13.51**	**13.78**
FP03	56	15.20	15.40	15.40	15.70	15.50	15.80	15.70	16.00	15.80	16.10	15.90	16.20	**13.76**	**14.08**
FP04	66	16.00	16.20	16.10	16.30	16.20	16.50	16.30	16.60	16.40	16.70	16.50	16.80	**14.58**	**15.22**
FP05	81	17.30	17.50	17.40	17.70	17.50	17.80	17.60	18.00	17.80	18.10	18.00	18.20	**16.02**	**17.81**
FP06	52	14.10	14.30	14.20	14.40	14.30	14.60	14.40	14.70	14.50	14.80	14.70	14.90	**12.26**	**12.77**
FP07	77	15.50	15.80	15.70	16.00	15.80	16.10	15.90	16.20	16.00	16.20	16.10	16.40	**14.02**	**14.45**
FP08	102	18.20	18.50	18.40	18.70	18.50	18.80	18.60	18.90	18.70	19.00	18.80	19.10	**15.51**	**15.98**
FP09	152	20.50	20.80	20.60	20.90	20.80	21.10	20.90	21.20	21.00	21.30	21.20	21.50	**18.73**	**19.87**
FP10	201	22.60	22.90	22.70	23.00	22.80	23.10	22.90	23.20	23.10	23.40	23.20	23.50	**21.51**	**22.63**
FP11	17	8.98	9.02	9.02	9.10	8.99	9.20	9.01	9.30	9.00	9.20	9.10	9.40	**8.95**	**8.96**
FP12	21	9.20	9.33	9.40	9.46	9.50	9.52	9.30	9.35	9.40	9.45	9.38	9.45	**9.09**	**9.13**

For test cases with smaller customer sizes (e.g., FP01, FP02, FP11, and FP12), the performance difference between Loegised and the algorithm with one module ablated is not significant. These test cases are characterized by relatively small problem sizes and a limited search space, which allows the genetic algorithm to quickly find approximate optimal solutions even without the optimization modules. Therefore, for these simpler test cases, the effect of the three optimization modules on the solution quality is not prominent. Among these simple test cases, the impact of the population initialization optimization module is relatively small, especially notable for its minor effect. Given the small problem size, the initial solution does not significantly influence the final result. However, with module ablation, there is a slight decrease in performance. For example, in the FP01 test case, the difference between Loegised$-_1$ (removing the population initialization optimization module) and Loegised$-_2$ (removing the dynamic operator parameter adjustment module) is small. This indicates that for simpler problems, the initial search capability of the genetic algorithm is strong, and the role of optimization modules in performance improvement is relatively limited.

As the complexity of the test cases increases, such as the medium-sized cases FP03, FP06, and FP07, the performance differences from ablation modules begin to emerge. In these test cases, ablation of the optimization modules leads to performance degradation, particularly in terms of BST and AVG results, with the ablation magnitude gradually increasing. These test cases are characterized by an expanding search space, and the impact of ablating optimization modules on the algorithm's performance becomes more apparent. In these medium-complexity cases, the roles of the dynamic operator parameter adjustment module and the local optimum escape mechanism module begin to stand out. In particular, the ability to dynamically adjust the crossover and mutation rates provides greater flexibility in adjusting the search strategy. In FP06, Loegised$-_1$ (removing the population initialization optimization module) performs worse than Loegised$-_2$ (removing the dynamic operator adjustment module), where the latter shows a more significant performance decline, indicating the importance of dynamic operator parameter adjustment for stability and convergence. Furthermore, the local optimum escape mechanism helps improve solution quality by preventing the algorithm from getting stuck in local optima. This is particularly evident in FP07, where Loegised$-_3$ performs better than the version without this mechanism.

For high-complexity test cases (e.g., FP08, FP09, and FP10), the performance differences due to module ablation become more significant. These test cases typically involve larger customer sizes and more complex delivery paths, making the algorithm more reliant on the optimization modules to search the solution space. In these complex test cases, the role of optimization modules becomes indispensable, and ablation of any module results in a noticeable performance drop. This effect is especially pronounced when two modules are simultaneously ablated. In these high-complexity test cases, the contributions of all three optimization modules become particularly important, with the population initialization optimization module playing an especially crucial role. In the FP09 test case, Loegised achieved a BST of 18.73 and AVG of 19.87, whereas Loegised$-_1$ (removing the population initialization optimization module) resulted in a BST of 20.50 and AVG of 21.00, highlighting the critical role of this module in improving search efficiency. As the customer size increases, the contributions

 doi:10.20944/preprints202505.1104.v1

of the dynamic operator parameter adjustment module and the local optimum escape mechanism module become more prominent in avoiding local optima and adjusting search strategies. In FP10, the performance drops sharply when these two modules are removed, indicating their importance in optimizing the search process for large-scale problems.

4.4.3. Comparison Results of Loegised with Baseline Methods

Table 5 shows the comparison of the Loegised method with six baseline methods (BB, CPM, SA, ALNS, IPGA, and HSFLA) on multiple test cases. The parameters of the baseline algorithms are listed in Table 4. Specifically, the table includes the following parameters: N_{POP} (population size), t_p (runtime), T_0 (initial temperature), λ (reaction parameter), q (cooling factor), and L (chain length). Since the MDVCP-DR problem involves various complex constraints, existing strategies are unable to ensure solution feasibility. Therefore, the initial solution generation and encoding-decoding process for all baseline algorithms are implemented using the method proposed in this paper. In the SA algorithm, the initial temperature is set to $T_0 = 1000$, but due to the optimization objective in [42] being different from that considered in this paper, the original parameters are no longer applicable. To adapt to the objective in this paper, the cooling factor has been adjusted to 0.01 as per [43]. Additionally, since the encoding method in this paper does not include the information about whether the customer is served by a drone or a vehicle, the neighborhood structure in ALNS does not explicitly specify whether the path is a drone path or a vehicle path. For IPGA, based on the study by Zhou et al. [26], it uses a sequence-breakpoint encoding method, whereas this paper uses only sequence encoding, so only the adjustments related to the sequence are applied in IPGA. Other parameters and settings remain consistent with the source literature. In the specific experiment, the vehicle and drone payloads are set to $L_m = 5$ kg and the time limit is set to $L_{time} = 0.5$ h. The results from the comparison between BB, CPM, IPGA, SA, ALNS, and HSFLA algorithms on FP01–FP12 are used for further analysis.

Table 4. Parameter Setting of Baseline Algorithms

Method	t_p/s	N_{POP}	T_0	λ	q	L
SA	n	1	0.01	-	0.90	2000
ALNS	n	1	0.01	0.9	0.98	1
IPGA	n	100	-	-	-	-
HSFLA	n	72	-	-	-	-
BB	n	1	-	-	-	-
CPM	n	1	-	-	-	-

Table 5. Simulation result of different algorithm with $L_m = 5$ kg, $L_{time} = 0.5$ h

Test Case	Original Test Case	Customer Size	BB		CPM		SA		ALNS		IPGA		HSFLA		Loegised	
			BST	AVG	BST	AVG	BST	AVG	BST	AVG	BST	AVG	BST	AVG	BST	AVG
FP01	A-n32-k05	33	13.20	13.60	13.30	13.65	13.38	13.57	13.33	13.75	15.53	16.36	13.13	13.36	**12.90**	**13.32**
FP02	A-n45-k06	46	13.85	14.15	13.70	13.95	13.80	14.69	14.15	14.89	17.80	18.71	13.52	13.77	**13.51**	**13.78**
FP03	A-n55-k09	56	14.12	15.13	14.20	15.12	14.13	15.05	13.98	15.29	18.85	20.43	13.89	14.23	**13.76**	**14.08**
FP04	A-n65-k09	66	16.78	17.45	16.85	17.30	15.58	16.53	15.04	16.32	20.84	22.04	14.76	15.39	**14.58**	**15.22**
FP05	A-n80-k10	81	21.85	22.30	21.55	22.60	19.54	20.89	19.06	20.95	28.90	31.22	16.24	17.90	**16.02**	**17.81**
FP06	CMT01	52	13.45	13.80	14.30	15.60	13.04	13.41	13.12	13.42	14.99	15.40	12.36	12.67	**12.26**	**12.77**
FP07	CMT02	77	16.50	17.85	16.80	17.50	15.22	16.08	15.65	16.32	19.79	20.27	14.21	14.59	**14.02**	**14.45**
FP08	CMT03	102	20.90	21.25	21.60	21.95	17.38	17.86	17.12	17.70	20.94	21.59	15.75	16.16	**15.51**	**15.98**
FP09	CMT04	152	28.50	30.90	29.20	30.40	22.85	23.71	21.46	22.76	28.72	29.90	19.36	20.24	**18.73**	**19.87**
FP10	CMT05	201	31.40	32.85	31.90	33.25	23.85	24.50	25.86	27.53	33.64	34.70	21.57	22.44	**21.51**	**22.63**
FP11	P-n016-k08	17	8.95	9.04	8.95	9.05	8.96	9.04	8.95	9.04	9.27	9.40	**8.95**	**8.96**	**8.95**	**8.96**
FP12	P-n020-k02	21	9.10	9.35	9.15	9.30	9.16	9.33	9.10	9.41	9.59	9.85	**9.09**	9.21	**9.09**	**9.13**

Table 5 presents the experimental results comparing the Loegised method with the other six baseline methods (BB, CPM, SA, ALNS, IPGA, and HSFLA) across multiple test cases. The comparison clearly shows that the Loegised method consistently outperforms all other methods, especially in test cases with larger customer sizes and higher problem complexities, where the method demonstrates outstanding global search capability, fast convergence, and high solution quality. For test cases with smaller customer sizes, such as FP01 (33 customers), FP02 (46 customers), and FP11 (17 customers), the advantages of the Loegised method are still evident. Despite the relatively small search space

 doi:10.20944/preprints202505.1104.v1

in these cases, the performance differences across all algorithms are minimal, but the BST and AVG values for Loegised remain lower than those for other baseline methods. For example, in the FP01 test case, Loegised achieved a BST of 12.90 and an AVG of 13.32, outperforming SA (13.38, 13.57) and ALNS (13.33, 13.75) with superior performance. Similarly, in test cases like FP02 and FP11, Loegised consistently maintained lower solution values, reflecting its strong solution quality and stability. Notably, BB and CPM also performed well in small-scale cases, particularly in FP01 and FP02, where their BST and AVG values were better than most heuristic algorithms. In FP01, BB's BST was 13.20 and AVG was 13.60, while CPM's BST was 13.30 and AVG was 13.65, outperforming SA (13.38, 13.57) and ALNS (13.33, 13.75), showing the clear advantage of exact algorithms for small-scale problems. The same trend was observed in FP02, where BB and CPM produced significantly lower solution values compared to heuristic algorithms, further proving the superiority of exact algorithms for small-scale problems.

As the problem size increases, especially in medium-sized cases such as FP03 (56 customers), FP06 (52 customers), and FP07 (77 customers), the advantages of Loegised begin to emerge more clearly. In FP03, Loegised achieved a BST of 13.76 and an AVG of 14.08, which were significantly better than ALNS (13.98, 15.29) and IPGA (18.85, 20.43). Moreover, although BB and CPM still performed relatively well, their BST and AVG values gradually increased as the problem size grew. In FP03, BB's BST was 14.20 and AVG was 14.40, while CPM's BST was 14.10 and AVG was 14.50, showing a noticeable gap in solution quality compared to Loegised, especially in medium-complexity problems. In FP06 and FP07, Loegised continued to excel, with particularly impressive performance in FP07 (77 customers), where Loegised achieved a BST of 14.02 and an AVG of 14.45, outperforming BB (16.50, 17.85) and CPM (16.80, 17.50). This further validates the advantage of Loegised in medium-sized problems, particularly in the presence of complex constraints and a large solution space, where heuristic algorithms can converge more quickly and obtain high-quality solutions.

In high-complexity test cases (such as FP08, FP09, and FP10), the advantages of Loegised become even more evident. These test cases involve larger customer sizes and broader solution spaces, where traditional algorithms often perform poorly in terms of convergence speed and solution quality. In the FP09 (152 customers) and FP10 (201 customers) test cases, Loegised achieved a BST of 18.73 and an AVG of 19.87, and a BST of 21.51 and an AVG of 22.63, respectively, while SA and ALNS showed significant performance drops. For example, in FP09, SA's BST was 22.85 and AVG was 23.71, while ALNS's BST was 21.46 and AVG was 22.76, all falling behind Loegised's performance. In large-scale test cases, BB and CPM's computational complexity limited their performance, with BB's BST in FP09 being 28.50 and AVG being 30.90, and CPM's BST being 29.20 and AVG being 30.40, all significantly lagging behind Loegised.

The comparison shows that while BB and CPM demonstrate strong performance in small-scale problems, heuristic algorithms like Loegised become more advantageous as the problem size increases, especially in medium- and large-scale problems. Overall, the Loegised method consistently performs well across all test cases, especially in high-complexity problems, where its fast convergence and high solution quality significantly outperform exact algorithms.

4.4.4. Sensitivity Analysis

To explore the sensitivity of the Loegised algorithm under different drone parameters, this experiment tests the algorithm's performance with various combinations of drone endurance and payload capacity limits. We selected the large-scale test case FP08, which contains 102 customers, and conducted 10 independent runs for each combination of L_{m} and L_{time}. The average result (AVG) for each experiment was recorded. Table 6 lists the AVG values for each set of experiments, with the best result in each experiment highlighted in bold.

From Table 6, it can be seen that the Loegised algorithm consistently exhibits optimal performance across all drone parameter configurations. For example, in Experiment 1 ($L_{\mathrm{m}} = 5\,\mathrm{kg}$, $L_{\mathrm{time}} = 0.5\,\mathrm{h}$), Loegised achieved an AVG of 15.76, significantly outperforming HSFLA (16.09), SA (17.09), ALNS (16.78), and IPGA (21.18). This optimized performance demonstrates the strong sensitivity advantage

 doi:10.20944/preprints202505.1104.v1

of the Loegised method to different drone parameter configurations. Particularly, when handling different combinations of endurance and payload capacity, Loegised consistently generates optimal task scheduling solutions, improving delivery efficiency and reducing total delivery time.

Table 6. Results of different algorithm on FP08 under different drone parameter

Experiment ID	L_m/kg	L_{time}/h	AVG						
			BB	CPM	SA	ALNS	IPGA	HSFLA	Loegised
1	5	0.5	20.9	21.6	17.09	16.78	21.18	16.09	**15.76**
2	5	0.75	20.5	21.4	16.02	15.94	18.98	15.58	**15.05**
3	5	1	20.2	21.2	15.93	15.88	18.36	15.28	**14.60**
4	5	1.25	20.1	21.0	15.73	15.73	18.37	15.28	**14.73**
5	5	1.5	19.8	20.8	15.90	15.93	18.35	15.09	**14.55**
6	6	0.5	20.6	21.5	17.18	16.65	21.43	16.00	**15.45**
7	6	0.75	20.4	21.3	16.07	15.96	18.66	15.11	**14.70**
8	6	1	20.2	21.1	15.71	15.55	18.05	15.09	**14.82**
9	6	1.25	20.0	20.9	15.51	15.61	17.97	14.96	**14.55**
10	6	1.5	19.8	20.7	15.57	15.48	17.92	15.06	**14.78**
11	7	0.5	20.4	21.3	16.83	16.86	21.06	16.17	**15.63**
12	7	0.75	20.3	21.2	15.88	16.01	18.46	15.21	**14.80**
13	7	1	20.1	21.0	15.54	15.48	17.99	14.87	**14.20**
14	7	1.25	20.0	20.9	15.23	15.18	17.70	14.91	**14.31**
15	7	1.5	19.8	20.7	15.28	15.32	17.79	14.90	**14.14**
16	8	0.5	20.6	21.5	17.24	16.75	20.94	15.74	**15.46**
17	8	0.75	20.4	21.3	16.12	15.71	18.26	15.09	**14.70**
18	8	1	20.2	21.1	15.45	15.45	18.05	14.79	**14.23**
19	8	1.25	20.1	21.0	15.26	15.29	17.90	14.68	**14.12**
20	8	1.5	19.9	20.8	15.27	15.14	17.62	14.75	**14.24**
21	9	0.5	20.5	21.4	17.15	16.68	20.80	15.85	**15.55**
22	9	0.75	20.3	21.2	16.09	16.01	18.30	15.16	**14.60**
23	9	1	20.1	21.0	15.36	15.62	18.06	15.01	**14.75**
24	9	1.25	19.9	20.8	15.29	15.32	17.49	14.78	**14.72**
25	9	1.5	19.8	20.7	14.98	14.87	17.41	14.67	**14.13**

Further analysis shows that even under extreme conditions, such as higher payload and endurance times, Loegised maintains higher stability and adaptability. For instance, in Experiment 25 ($L_m = 9\,kg$, $L_{time} = 1.5\,h$), Loegised achieved an AVG of 14.13, showing a noticeable improvement compared to other methods (such as HSFLA's 14.67, SA's 14.98, ALNS's 14.87, and IPGA's 17.41), further validating the advantage of Loegised when facing complex constraint conditions.

5. Conclusion

This paper addresses the complex Vehicle-Multi-Drone Cooperative Delivery Problem with Delivery Restrictions, which is characterized by multi-objective optimization, stringent spatial-temporal constraints, and high-dimensional solution spaces. To tackle the limitations of traditional optimization approaches—particularly genetic algorithms—in handling stagnation, poor initialization, and parameter sensitivity, we propose Loegised, a novel logistics scheduling framework that integrates Large Language Models into the core of the genetic algorithm pipeline. Specifically, Loegised incorporates a Cognitive Initialization Module to enhance population quality, a Dynamic Operator Parameter Adjustment Module to adaptively balance exploration and exploitation, and a Local Optimum Escape Mechanism to maintain search diversity. Extensive experiments conducted on MDVCP-DR benchmark datasets demonstrate that Loegised consistently outperforms state-of-the-art methods, including Gurobi, Simulated Annealing, ALNS, and IPGA. Our method achieves superior performance in both solution quality and computational efficiency, particularly in large-scale scenarios, validating the effectiveness and scalability of the LLM-guided optimization strategy.

 doi:10.20944/preprints202505.1104.v1

In future work, we aim to further extend the Loegised framework in three key directions. First, we plan to investigate multi-agent coordination mechanisms under real-time dynamic task arrivals and environmental changes, enabling our system to better support online decision-making. Second, to enhance scalability, we will explore parallel and distributed implementations of Loegised, leveraging GPU-based computation and federated optimization architectures. Third, we intend to incorporate multi-modal inputs, such as satellite maps and sensor data, into the LLM prompt engineering process to improve semantic representation of constraints and enhance generalization across different operational contexts. Additionally, we will explore hybrid neuro-symbolic integration, combining LLM-generated symbolic plans with neural reinforcement learning agents to achieve deeper situational awareness and adaptive learning in real-world logistics systems. These extensions will further elevate the practical utility and theoretical depth of LLM-empowered evolutionary optimization.

Author Contributions: Methodology, M.G.; Software, A.C.; Validation, M.G. and A.C.; Visualization, A.C.; Writing—original draft, M.G.; Writing—review and editing, A.C. All authors have read and agreed to the the published version of the manuscript.

Informed Consent Statement: Informed consent was obtained from all subjects involved in the study.

Data Availability Statement: Public data warehouse. All source code and data are publicly available at https://doi.org/10.5281/zenodo.15176124.

Funding: This research received no external funding.

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

References

1. Murray, C.C.; Chu, A.G. The flying sidekick traveling salesman problem: Optimization of drone-assisted parcel delivery. *Transportation Research Part C: Emerging Technologies* **2015**, *54*, 86–109.
2. Wen, X.; Wu, G. Heterogeneous multi-drone routing problem for parcel delivery. *Transportation Research Part C: Emerging Technologies* **2022**, *141*, 103763.
3. Shi, Y.; Lin, Y.; Lim, M.K.; Tseng, M.L.; Tan, C.; Li, Y. An intelligent green scheduling system for sustainable cold chain logistics. *Expert Systems with Applications* **2022**, *209*, 118378.
4. Wang, C.L.; Wang, Y.; Zeng, Z.Y.; Lin, C.Y.; Yu, Q.L. Research on logistics distribution vehicle scheduling based on heuristic genetic algorithm. *Complexity* **2021**, *2021*, 8275714.
5. Mouha, R.A.R.A.; et al. Internet of things (IoT). *Journal of Data Analysis and Information Processing* **2021**, *9*, 77.
6. Soori, M.; Arezoo, B.; Dastres, R. Internet of things for smart factories in industry 4.0, a review. *Internet of Things and Cyber-Physical Systems* **2023**, *3*, 192–204.
7. Laghari, A.A.; Wu, K.; Laghari, R.A.; Ali, M.; Khan, A.A. A review and state of art of Internet of Things (IoT). *Archives of Computational Methods in Engineering* **2021**, pp. 1–19.
8. Korteling, J.H.; van de Boer-Visschedijk, G.C.; Blankendaal, R.A.; Boonekamp, R.C.; Eikelboom, A.R. Human-versus artificial intelligence. *Frontiers in artificial intelligence* **2021**, *4*, 622364.
9. Jiang, Y.; Li, X.; Luo, H.; Yin, S.; Kaynak, O. Quo vadis artificial intelligence? *Discover Artificial Intelligence* **2022**, *2*, 4.
10. Zhang, C.; Lu, Y. Study on artificial intelligence: The state of the art and future prospects. *Journal of Industrial Information Integration* **2021**, *23*, 100224.
11. Yin, Y.; Qing, L.; Wang, D.; Cheng, T.; Ignatius, J. Exact solution method for vehicle-and-drone cooperative delivery routing of blood products. *Computers & Operations Research* **2024**, *164*, 106559.
12. Zhang, R.; Dou, L.; Xin, B.; Chen, C.; Deng, F.; Chen, J. A review on the truck and drone cooperative delivery problem. *Unmanned Systems* **2024**, *12*, 823–847.
13. Wang, D.; Hu, P.; Du, J.; Zhou, P.; Deng, T.; Hu, M. Routing and scheduling for hybrid truck-drone collaborative parcel delivery with independent and truck-carried drones. *IEEE Internet of Things Journal* **2019**, *6*, 10483–10495.
14. Peng, Y.; Zhu, W.; Yu, D.Z.; Liu, S.; Zhang, Y. Multi-Depot Electric Vehicle–Drone Collaborative-Delivery Routing Optimization with Time-Varying Vehicle Travel Time. *Vehicles* **2024**, *6*, 1812–1842.
15. Agatz, N.; Bouman, P.; Schmidt, M. Optimization approaches for the traveling salesman problem with drone. *Transportation Science* **2018**, *52*, 965–981.

 doi:10.20944/preprints202505.1104.v1

16. Gonzalez-R, P.L.; Canca, D.; Andrade-Pineda, J.L.; Calle, M.; Leon-Blanco, J.M. Truck-drone team logistics: A heuristic approach to multi-drop route planning. *Transportation Research Part C: Emerging Technologies* **2020**, *114*, 657–680.
17. Das, D.N.; Sewani, R.; Wang, J.; Tiwari, M.K. Synchronized truck and drone routing in package delivery logistics. *IEEE Transactions on Intelligent Transportation Systems* **2020**, *22*, 5772–5782.
18. Luo, Z.; Poon, M.; Zhang, Z.; Liu, Z.; Lim, A. The multi-visit traveling salesman problem with multi-drones. *Transportation Research Part C: Emerging Technologies* **2021**, *128*, 103172.
19. Gu, R.; Poon, M.; Luo, Z.; Liu, Y.; Liu, Z. A hierarchical solution evaluation method and a hybrid algorithm for the vehicle routing problem with drones and multiple visits. *Transportation Research Part C: Emerging Technologies* **2022**, *141*, 103733.
20. DU, M.; LUO, J.; LI, B. Research on cooperative delivery route optimization of vehicle-carried drones based on multi-depot. *Systems Engineering* **2021**, *39*, 90–98.
21. Kuo, R.; Lu, S.H.; Lai, P.Y.; Mara, S.T.W. Vehicle routing problem with drones considering time windows. *Expert Systems with Applications* **2022**, *191*, 116264.
22. Luo, Q.; Wu, G.; Ji, B.; Wang, L.; Suganthan, P.N. Hybrid multi-objective optimization approach with Pareto local search for collaborative truck-drone routing problems considering flexible time windows. *IEEE Transactions on Intelligent Transportation Systems* **2021**, *23*, 13011–13025.
23. Peng, Y.; Li, Y. Optimization of truck-drone collaborative distribution route considering impact of epidemic. *China Journal of Highway and Transport* **2020**, *33*, 73–82.
24. Yan, R.; Chen, L.; Zhu, X.; Tian, H.; Wen, Y.; Zhang, Q. Research on vehicle routing problem with truck and drone considering regional restriction. *Chinese Journal of Management Science* **2022**, *30*, 144–155.
25. Eusuff, M.; Lansey, K.; Pasha, F. Shuffled frog-leaping algorithm: a memetic meta-heuristic for discrete optimization. *Engineering optimization* **2006**, *38*, 129–154.
26. Zhou, H.; Song, M.; Pedrycz, W. A comparative study of improved GA and PSO in solving multiple traveling salesmen problem. *Applied Soft Computing* **2018**, *64*, 564–580.
27. Duan, H.; Li, X.; Lu, Q.; Lin, S. Hybrid Shuffled Frog Leaping Algorithm for Solving Vehicle-Drone Cooperative Delivery Problem. *J. Zhejiang Univ. Eng. Sci* **2024**, *58*, 2258–2269.
28. Geng, M.; Wang, S.; Dong, D.; Wang, H.; Li, G.; Jin, Z.; Mao, X.; Liao, X. Large language models are few-shot summarizers: Multi-intent comment generation via in-context learning. In Proceedings of the Proceedings of the 46th IEEE/ACM International Conference on Software Engineering, 2024, pp. 1–13.
29. Geng, R.; Geng, M.; Wang, S.; Wang, H.; Lin, Z.; Dong, D. Mitigating Sensitive Information Leakage in LLMs4Code through Machine Unlearning. *arXiv preprint arXiv:2502.05739* **2025**.
30. Ugare, S.; Suresh, T.; Kang, H.; Misailovic, S.; Singh, G. Improving llm code generation with grammar augmentation. *arXiv preprint arXiv:2403.01632* **2024**.
31. Feng, Z.; Zhang, Y.; Li, H.; Liu, W.; Lang, J.; Feng, Y.; Wu, J.; Liu, Z. Improving llm-based machine translation with systematic self-correction. *arXiv preprint arXiv:2402.16379* **2024**.
32. Yang, Z.; Liu, F.; Yu, Z.; Keung, J.W.; Li, J.; Liu, S.; Hong, Y.; Ma, X.; Jin, Z.; Li, G. Exploring and unleashing the power of large language models in automated code translation. *arXiv preprint arXiv:2404.14646* **2024**.
33. Dong, Y.; Ding, J.; Jiang, X.; Li, G.; Li, Z.; Jin, Z. Codescore: Evaluating code generation by learning code execution. *arXiv preprint arXiv:2301.09043* **2023**.
34. Ahmed, T.; Pai, K.S.; Devanbu, P.; Barr, E. Automatic semantic augmentation of language model prompts (for code summarization). In Proceedings of the Proceedings of the IEEE/ACM 46th International Conference on Software Engineering, 2024, pp. 1–13.
35. Li, B.; Sun, Z.; Huang, T.; Zhang, H.; Wan, Y.; Li, G.; Jin, Z.; Lyu, C. IRCoCo: Immediate Rewards-Guided Deep Reinforcement Learning for Code Completion. *arXiv preprint arXiv:2401.16637* **2024**.
36. Zhang, K.; Zhang, H.; Li, G.; Li, J.; Li, Z.; Jin, Z. Toolcoder: Teach code generation models to use api search tools. *arXiv preprint arXiv:2305.04032* **2023**.
37. Wang, Z.; Li, J.; Li, G.; Jin, Z. ChatCoder: Chat-based Refine Requirement Improves LLMs' Code Generation. *arXiv preprint arXiv:2311.00272* **2023**.
38. Dong, Y.; Jiang, X.; Jin, Z.; Li, G. Self-collaboration code generation via ChatGPT (2023). *arXiv preprint arXiv:2304.07590*.
39. Microsoft Research Asia. Parrot: Optimizing logistics with semantic variables using large language models **2024**.
40. Aliyun Developer Community. The potential of large models in the logistics industry **2024**.

 doi:10.20944/preprints202505.1104.v1

41. Mongaillard, T.; Lasaulce, S.; Hicheur, O.; Zhang, C.; Bariah, L.; Varma, V.S.; Zou, H.; Zhao, Q.; Debbah, M. Large language models for power scheduling: A user-centric approach. In Proceedings of the 2024 22nd International Symposium on Modeling and Optimization in Mobile, Ad Hoc, and Wireless Networks (WiOpt). IEEE, 2024, pp. 321–328.
42. Liu, W.; Li, W.; Zhou, Q.; Die, Q. "Drone-Vehicle" Distribution Routing Optimization Model. *Journal of Transportation Systems Engineering and Information Technology* **2021**, *21*, 176–186.
43. Guohua, W.; Ni, M.; Binjie, X.; et al. The cooperative delivery of multiple vehicles and multiple drones based on adaptive large neighborhood search. *Control and Decision* **2023**, *38*, 201–210.
44. Lawler, E.L.; Wood, D.E. Branch-and-bound methods: A survey. *Operations research* **1966**, *14*, 699–719.
45. Kelley, Jr, J.E. The cutting-plane method for solving convex programs. *Journal of the society for Industrial and Applied Mathematics* **1960**, *8*, 703–712.