A Coordinated Vehicle–Drone Arc Routing Approach Based on Improved Adaptive Large Neighborhood Search

Guohua Wu 1, Kexin Zhao 1, Jiaqi Cheng 1 and Manhao Ma 2,*

1 School of Traffic & Transportation Engineering, Central South University, Changsha 410075, China; guohuawu@csu.edu.cn (G.W.); kx_zhao@csu.edu.cn (K.Z.); 204207032@csu.edu.cn (J.C.)
2 College of Systems Engineering, National University of Defense Technology, Changsha 410073, China
* Correspondence: mhma@nudt.edu.cn

Abstract: Through urban traffic patrols, problems such as traffic congestion and accidents can be found and dealt with in time to maintain the stability of the urban traffic system. The most common way to patrol is using ground vehicles, which may be inflexible and inefficient. The vehicle–drone coordination maximizes utilizing the flexibility of drones and addresses their limited battery capacity issue. This paper studied a vehicle–drone arc routing problem (VD-ARP), consisting of one vehicle and multiple drones. Considering the coordination mode and constraints of the vehicle–drone system, a mathematical model of VD-ARP that minimized the total patrol time was constructed. To solve this problem, an improved, adaptive, large neighborhood search algorithm (IALNS) was proposed. First, the initial route planning scheme was generated by the heuristic rule of “Drone-First, Vehicle-Then”. Then, several problem-based neighborhood search strategies were embedded into the improved, adaptive, large neighborhood search framework to improve the quality of the solution. The superiority of IALNS is verified by numerical experiments on instances with different scales. Several critical factors were tested to determine the effects of coordinated traffic patrol; an example based on a real road network verifies the feasibility and applicability of the algorithm.

Keywords: vehicle–drone; arc routing problem; traffic patrol; routing optimization; adaptive large neighborhood search

1. Introduction

With the continuous increase in car ownership in urban areas, congestion, traffic accidents, and other adverse factors seriously affect the operation of the urban traffic system [1], which is a huge challenge for traffic management departments. In order to maintain the stability of the urban traffic system, a traffic patrol of the urban road network is essential. At present, there are problems such as poor pertinence, inflexibility, and untimely monitoring in traditional police patrols using ground vehicles. By developing civil drones and making full use of their flexibility, mobility, and timeliness, the efficiency of urban traffic patrols can be further improved [2]. However, the widespread application of drones is limited by their battery capacities [3].

The combination of vehicle and drone gives full range to their advantages, and makes up for their shortcomings [4]. Drones are less limited by ground traffic, and can perform better in places that are difficult for vehicles to reach. Meanwhile, when not visiting tasks, the drones stay on the vehicle and the battery can be replaced to prepare for the next flight. In the past few years, research on coordinated vehicle–drone routing was mostly applied to parcel deliveries [5]. For example, Murray and Chu [4] first proposed the flying sidekick traveling salesman problem (FSTSP) in 2015, to study the parcel delivery problem considering a vehicle and a drone. In this problem, the drone can only visit one customer after launching from the vehicle each time, and the vehicle cannot stop in place to wait and recover the drone. Agatz et al. [6] propose the traveling salesman problem with drone
(TSP-D), and introduce a heuristic approach to solve it. Compared with FSTSP, in the TSP-D, the drone can be launched and recovered at the same node. Based on the above two studies, a large number of studies emerged considering more complex constraints, such as time windows [7], different objective functions [8,9], heterogeneous drones [10], and simultaneous pickup and delivery [11]. Meanwhile, heuristic methods [6,12], a simulated annealing algorithm [13], a genetic algorithm [14], and variable neighborhood search algorithm [15] were designed to solve these problems. However, although some of the above studies on coordinated vehicle-drone parcel delivery aim at visiting customer nodes, they are still variants of the vehicle routing problem (VRP) [16]. This paper studied patrol routing of the urban road network, and focused on segments of the road network, that is, the road network itself. This problem can be defined as an extension of the vehicle–drone arc routing problem [17].

The vehicle–drone arc routing problem can be applied to many scenarios, such as urban traffic patrol and road snow clearing, and has great potential for research and application [18]. However, there are few studies on arc routing of vehicle–drone coordination. Among the research available, there are only two studies on the coordinated vehicle–drone arc routing problem. Liu et al. [19] apply the combination mode of a vehicle and a drone to high-voltage powerline inspection, in which the drone inspects the powerline and the vehicle is used for launching/recovering the drone at a specific rendezvous. Vehicle and drone traveled on different maps, where the planning of vehicle path is still a VRP. Luo et al. [20] first propose and solve a traffic patrol model with a vehicle and a drone, and they consider both arc tasks and point tasks. However, the above two studies only consider patrolling with the combination of one vehicle and one drone, which is inefficient compared with that of one vehicle and multiple drones. Hence, this study focuses on a more difficult problem, which allows one vehicle and multiple drones to patrol cooperatively. Besides, to be more realistic, we consider that the drones fly without following the road network. We name the studied problem the vehicle–drone arc routing problem, consisting of a vehicle and multiple drones (VD-ARP).

It is a great challenge to solve VD-ARP and give the routing planning scheme of the vehicle and the drones. First, considering the traffic patrol requirements, the vehicle must move along the road network, and needs to be coordinated with multiple drones, which makes it very difficult to handle the routing constraints. Second, the encoding method of the vehicle route and drone routes may affect the complexity of solving the problem, while it is a challenge to design an elegant encoding of the target edges. Third, it is worth considering how to design the search framework and effective neighborhood search strategies, so that the algorithm can meet the time and efficiency requirements of an urban traffic patrol. In this paper, all the above challenges were properly addressed. Specifically, an improved, adaptive, large neighborhood search algorithm (IALNS) was designed. IALNS searched the solution space and obtained new alternative solutions through the adaptive large neighborhood search mechanism, and received the new solution, according to the Metropolis criterion in the simulated annealing algorithm. Moreover, a tabu table was set up in the search process, in order to reduce the repeated search in a short period of time.

The major contributions of this paper are as follows.

- We proposed a vehicle–drone arc routing problem considering a vehicle and multiple drones (VD-ARP), and considered a more realistic scenario in which drones fly outside of the road network in addition to visiting tasks. This was rarely investigated in the past.
- To solve VD-ARP, we designed an improved, adaptive, large neighborhood search algorithm (IALNS). First, the task edges were encoded and the initial solution was generated by the “Drone-First, Vehicle-Then” heuristic method. Then, a set of problem-based neighborhood search strategies were designed. Finally, the simulated annealing mechanism, tabu table, and multiple neighborhood search strategies were embedded into the adaptive large neighborhood search framework to improve the solution.
Compared with the existing heuristic methods, IALNS can be applied to larger-scale tasks, and obtain an acceptable solution in a short time.

- The effectiveness and superiority of IALNS are verified based on the experiments of instances with different scales. In addition, a case study was carried out to show that IALNS provides a satisfactory traffic patrol routing scheme in a short time.

The remainder of this paper is structured as follows. Section 2 provides a brief review of related works and Section 3 describes the VD-ARP and gives a mathematical model. Section 4 details the proposed improved, adaptive, large neighborhood search algorithm, and Section 5 reports and discusses the experimental studies. Finally, Section 6 gives conclusions.

2. Related Work

In recent years, traffic patrols have become an important way to prevent and respond to emergencies in urban traffic management [21]. Through traffic patrols, it is possible to intervene in congestion and monitor vehicle violations, as well as road facilities and equipment. Steil et al. [22] study the expression, execution, evaluation, and engagement of patrol routing algorithms for accident-prone locations on roads, so as to provide acceptable patrol routes for organizations such as police agencies and emergency medical responders. Jalili et al. [23] design a set of comprehensive measures for highway patrol agencies to measure and improve the management effectiveness, in order to improve road network safety. Li et al. [24] propose a new evaluation standard to evaluate the effect of traffic patrols in emergency situations. Chawathe et al. [25] solve a problem of patrol route planning, aiming at minimizing the length of patrol routes, and maximizing coverage of patrol locations. There are also some studies related to traffic patrols with drones. Wang et al. [26] use drones to collect data for vehicle speed analyses in the road network. Munishkin et al. [27] and Wu et al. [28] propose an information transmission model and an adaptive data processing system for vehicles and drones respectively, providing theoretical support for a coordinated traffic patrol of vehicles and drones. The main contribution of this paper is to design a route planning algorithm for a coordinated vehicle–drone traffic patrol problem by taking the required road segments as patrol targets.

When the patrol targets of vehicles and drones are segments in the road network, this problem is described as a variant of the arc routing problem [17]. The arc routing problem was developed from graph theory [29], and the most widely studied arc routing problems include the Chinese postman problem (CPP) [30,31], rural postman problem (RPP) [32–34], and capacitated arc routing problem (CARP) [35,36]. Kwan [37] first proposed the CPP, and designed a deterministic solution method. The problem is described as traversing all edges of a graph, and minimizing the total distance. Lima et al. [38] solve the CPP based on a mixed graph through ant colony algorithm, which is proven to be an NP-hard problem. As for RPP, only a part of the edges in the graph should be the accesses. Hertz et al. [39] reconstruct the undirected RPP problem, and design a variety of heuristic methods to obtain high-quality solutions. Fernandez et al. [40] study a multi-depot RPP, in which each route starts and ends at the same depot, and develop a branch-and-cut algorithm. Zhang et al. [41] propose a multi-depot CARP of a large scale, and introduce an iterative improved heuristic algorithm to solve it. Babaee et al. [42] design a hybrid ant colony optimization algorithm to solve the multi-trip CARP, and apply it to urban solid waste management. With the widespread use of drones, the arc routing problem has been extended, since drones can fly between any two nodes without following the road network. Oh et al. [43] present the Chinese postman problem with drones, traversing all edges of the road network by drones. Campbell et al. [44] first define the drone arc routing problem (DARP), in which the drone can move directly between any two points in the graph, and the entire edge accessed through multiple drone flights. However, the above studies on arc routing are all for the route planning of a patrol platform (vehicles or drones), while this paper considers both vehicles and drones, which is a more complex combinatorial optimization problem.
In the last few years, research related to route planning combining vehicles and drones gradually emerged, and was applied to parcel deliveries \cite{45}. Murray and Chu \cite{4} and Agatz et al. \cite{6} first propose FSTSP and VRP-D, respectively, to enable a vehicle and a drone to visit customer nodes in parallel, and design heuristic methods to plan the vehicle and drone routes. In the literature review of Chung et al. \cite{5}, the research on various joint vehicle-drone operations, based on parcel delivery, is introduced in detail. However, to the best of our knowledge, there are few related studies on the arc routing problem for coordinated vehicle–drone operations. Luo et al. \cite{20} study an urban patrol problem considering a vehicle and a drone, and design a heuristic method to solve the problem. However, in this paper, we consider the combination of multiple drones and one vehicle, and assume that the drones can fly between any two nodes in the road network.

Table 1 presents the research status of representative parts of relevant works. Obviously, most studies consider road network scenarios. As for the research on routing optimization, there are two types of tasks, i.e., arc and point. It should be noted that most of the studies on routing optimization are based on one platform (vehicles or drones), while this paper focuses on the coordination of vehicles and drones. Moreover, different from \cite{20}, we consider the combination of one vehicle and multiple drones, which is more complex and flexible than that of one vehicle and one drone.

3. Problem Description

The proposed VD-ARP in this paper can be described as: a vehicle departs from the depot, carrying multiple drones, which can be launched and recovered several times from the vehicle, and the vehicle and the drones jointly access all target edges in the road network. Finally, the vehicle returns to the depot with all the drones. Once a drone accesses the required target edges in a flight, it needs to return to the vehicle. The vehicle also acts as an energy supply station for the drones to replace the battery. In this problem, some of the target edges are required to be accessed only by drones. To simplify the problem and facilitate the model, we consider the following assumptions:

- The launch or recovery operations of a drone must be carried out at a node (intersection), and the vehicle can perform multiple launch/recovery operations at one node;
- After the vehicle launches/recovers a drone, it can wait in place, or go to another node to recover the drone or launch another drone;
- The vehicle must move along the road network, while the drone routes are not restricted by the road network;
- The launch/recovery time of a drone is very short compared to the travel time on the road. Therefore, as in references \cite{6,19}, the time of launching/recovering a drone is incorporated into the travel time here.

| References       | Research Topic       | Vehicles | Drones | Objective | Task Type | Network |
|------------------|----------------------|----------|--------|-----------|-----------|---------|
| Jalili et al. \cite{23} | Traffic safety management | M        | -      | -         | -         | Yes     |
| Li et al. \cite{24}   | Evaluation system    | M        | -      | -         | -         | Yes     |
| Wang et al. \cite{26} | Data analysis        | M        | n      | Cost      | Arc       | Yes     |
| Kwan \cite{37}       | Routing optimization | 1        | -      | Cost      | Arc       | Yes     |
| Hertz et al. \cite{39} | Routing optimization | 1        | -      | Cost      | Arc       | Yes     |
| Babaee et al. \cite{42} | Routing optimization | M        | -      | Cost      | Arc       | Yes     |
| Campbell et al. \cite{44} | Routing optimization | -       | 1      | Cost      | Arc       | Yes     |
| Murray et al. \cite{4}      | Routing optimization | 1       | 1      | Time      | Point     | No      |
| Dorling et al. \cite{3}    | Routing optimization | M        | n      | Time/Cost | Point     | No      |
| Agatz et al. \cite{6}      | Routing optimization | 1       | 1      | Time      | Point     | Yes     |
| Hu et al. \cite{10}       | Routing optimization | 1       | n      | Time      | Point     | Yes     |
| Luo et al. \cite{20}      | Routing optimization | 1       | 1      | Time      | Point & Arc | Yes     |
| This Work             | Routing optimization | 1       | n      | Time      | Arc       | Yes     |
The urban road network is simplified as an undirected connected graph, represented by $G = (V, E)$. The set of nodes (intersections of the road network) is expressed as $V = \{1, 2, \ldots, n\}$, where $n$ is the number of nodes. The set of edges (road segments) is denoted as $E = \{e_{ij} = (i, j) | i, j \in V\}$, $e_{ij}$ for connecting nodes $i$ and $j$. The road network is undirected, so $e_{ij}$ and $e_{ji}$ represent the same edge. Since the vehicle moves along the road network, in order to distinguish the start nodes and end nodes of all edges that the vehicle passes through, the set $V_- \subset V$ represents the start node set, and $V_+ \subset V$ represents the end node set. Depots in both sets are represented by $v_0$ and $v_{PC}$, respectively. $T$ is the general target edge set, while $TD$ is the target edge set that can only be accessed by drones, $T, TD \subseteq E$. Figure 1 is a schematic diagram of the road network and target edges. In the figure, according to the above definition, $T = \{(13, 21),(11, 16),(15, 16)\}$ and $TD = \{(7, 8),(9, 14)\}$. In addition, other symbols and descriptions are shown in Table 2.

Table 2. Symbols and descriptions.

| Parameters | Descriptions |
|-----------|-------------|
| $D$       | Set of drones, $D = \{1, 2, \ldots, d\}$ |
| $A$       | Matrix variable to store information of target edges |
| $SVTE_i$  | Set of successively visited target edges in the $i$-th drone flight |
| $[l_{ij}, r_{ij}]$ | The $j$-th pair of launch and recovery nodes for the $i$-th target edge |
| $R$       | Vehicle route, $R = \{v_0, \ldots, v_{PC}\}$ |
| $l, SVTE_i, r$ | The drone route which visits the target edges in set $SVTE_i$, with $l$ for the launch node and $r$ for the recovery node |
| $w_{ij}$  | Length of edge $e_{ij}$ |
| $t_i$     | The time when the vehicle arrives at node $i$ |
| $t^k_i$   | The time when the $k$-th drone arrives at node $i$ |
| $e_i$     | The time interval required for the vehicle to wait for the recovery of the drones after arriving at the node $i$ |
| $v_{veh}$ | Vehicle speed |
| $v_D$     | Drone speed |
| $P$       | Maximum flight time of a drone |
| Decision variables | Descriptions |
| $x_{ij}$  | $x_{ij} = 1$ if the vehicle accesses the edge $e_{ij}$, otherwise, $x_{ij} = 0$ |
| $y^k_{ij}$| $y^k_{ij} = 1$ if the $f$-th flight of the $k$-th drone accesses the edge $e_{ij}$, otherwise, $y^k_{ij} = 0$ |
| $z^k_{ij}$| $z^k_{ij} = 1$ if the $f$-th flight of the $k$-th drone is launched from $i$ and recovered at $j$, otherwise, $z^k_{ij} = 0$ |
| $\alpha^k_i$ | Auxiliary variable, $\alpha^k_i = 0$ if the $k$-th drone is launched or recovered at $i$, otherwise, $\alpha^k_i = 1$ |
| $\alpha^{t,k}$ | Auxiliary variable, $\alpha^{t,k} = 1$ if the $k$-th drone has been launched at time $t$, otherwise, $\alpha^{t,k} = 0$ |

As mentioned above, the objective function consists of two parts: the time it takes for the vehicle to arrive at the depot, and the time spent waiting at the depot for all drones to return to the vehicle. Hence, the optimization objective of the patrol routing planning problem is the total patrol time, which can be written as

$$\min t = t_{PC} + \varepsilon_{PC}$$

where the first part is the time when the vehicle arrives at the depot $v_{PC}$, and $\varepsilon_{PC}$ represents the waiting time for the vehicle to recover all drones at the depot. When the vehicle is not the last to arrive at node $i$, $\varepsilon_i = \left(\max \{t^k_i \} - t_i \right)$, when the vehicle arrives last, $\varepsilon_i = 0$. It is worth noting that $t^k_i$ denote the time when the $k$-th drone arrives at node $i$, where $k \in D$. 

\[ t^k_i = t_{PC} + \varepsilon_{PC} \]
In addition, the main constraints that the problem should satisfy are as follows.

\[ \sum_{e_{ij} \in E} x_{ij,k} = \sum_{e_{ij} \in E} x_{k,ij}, \forall k \in V \]  

\[ \sum_{e_{ij} \in E} x_{0,ij} = 1, \forall i \in V + \]  

\[ \sum_{e_{ij} \in E} x_{ij,PC} = 1, \forall j \in V - \]  

\[ x_{ij} + x_{ji} + \sum_{e_{ij} \in E} (y_{ij}^{k_f} + y_{ji}^{k_f}) \geq 1, \forall e_{ij} \in T \]  

\[ \sum_{e_{ij} \in E} (y_{ij}^{k_f} + y_{ji}^{k_f}) \geq 1, \forall e_{ij} \in TD \]  

\[ \sum_{j \in V} z_{ij}^{k_f} = 1, \forall i \in V, k \in D \]  

\[ \sum_{i \in V} z_{ij}^{k_f} = 1, \forall j \in V, k \in D \]  

\[ \sum_{j \in V} z_{ij}^{k_f} \leq \sum_{s \in V} x_{is}, \forall i, j \in V, j \neq s \]  

\[ \sum_{i \in V} z_{ij}^{k_f} \leq \sum_{v \in V} x_{vj}, \forall i, j \in V, i \neq v \]  

\[ a_{ij}^{k_f} \left( \sum_{j \in V^+} y_{ij}^{k_f} \right) = 0, \forall i \in V, e_{ij} \in E \]  

\[ x_{ij} \left( \sum_{j \in V^+} y_{ij}^{k_f} \right) a_{ij}^{k_f} = 0, \forall i, j \in V, e_{ij} \in E \]  

\[ \sum_{k \in D} (a_{ij}^{k_f}) \leq d \]  

\[ t_j \geq t_i + w_{ij}/v^\text{Veh} - M(1 - x_{ij}), \forall i, j \in V, i \neq j \]  

\[ t_i^k \geq t_i^k + w_{ij}/v^D - M(1 - y_{ij}^{k_f}), \forall i, j \in V, i \neq j \]  

\[ t_i^k - t_i^j \leq P \]  

\[ \epsilon_{ij} = \begin{cases} m \max (t_{ij}, t_{ji}) & \text{if } i \neq j \\
0 & \text{if } i = j \end{cases} \]
\[ t_r - t_l^k \leq P \]  
(17)

\[ z_{i,r}^{k,f} = 1 \]  
(18)

Constraint (2) shows that for any node \( k \), the number of inward and outward arcs are the same, which means that the number of edges the vehicle enters node \( k \) and exits from node \( k \) are the same. Constraints (3) and (4) ensure that the vehicle starts from the depot only once, and eventually returns to the depot. Constraints (5) and (6) ensure that the two types of target edges are accessed at least once. Constraints (7) and (8) ensure that each drone flight has only one launch node and one recovery node. Constraints (9) and (10) indicate that the coordination relationship between the drones and the vehicle is associated with the nodes in the vehicle that launch and recover the drones, ensuring that the launch and recovery nodes of drones must be nodes in the vehicle route.

There are also strict restrictions on the order in which drones can be launched and recovered. A drone can just be launched only if it has never been launched before, or it has been recovered since the last flight. If the drone has not been recovered at node \( i \), it cannot be released from the vehicle. Constraints (11) and (12) stipulate that when the auxiliary variable \( a_{i,r}^k = 1 \), the \( k \)-th drone cannot be launched or recovered at node \( i \). In constraint (13), the number of drones flying at any time is limited to \( d \). The time constraints of the vehicle and drones are shown in constraints (14) and (15). Constraints (14) and (15) represent the earliest arrival time of the vehicle and drones at each node, respectively, where \( M \) is an extremely large positive number. Constraints (16) and (17), respectively, express the constraints of the maximum patrolling time when the vehicle and the drones arrive at node \( r \) first, where \( l \) and \( r \) meet constraint (18); that is, the launch and recovery nodes of the \( k \)-th drone in the \( f \)-th flight are \( l \) and \( r \), respectively.

4. Patrol Route Planning Approach

This section introduces the solution method for VD-ARP, namely the improved, adaptive, large neighborhood search algorithm (IALNS). It is mainly introduced from the following four parts: target edge encoding, initial solution generation, framework of IALNS algorithm, and neighborhood search strategies. All operations must satisfy the constraints described in Section 3.

4.1. Target Edge Encoding

Due to the limited battery capacity of drones, only a limited number of target edges can be accessed during a single flight. Consequently, we store the successively accessed target edges in the \( i \)-th drone flight in the set \( SVTE_i \). Assuming that the first drone flight is shown in Figure 2, the drone is launched from node 2, and recovered at node 8 after visiting target edges \( e_{53} \) and \( e_{64} \), then \( SVTE_1 = \{ e_{53}, e_{64} \} \). The first drone route can be recorded as 2, \( SVTE_1, 8 \).

![Figure 2. An example of a drone flight, where the numbers represent different nodes of the road network.](image)

The \( i \)-th pair of launch and recovery nodes for the \( i \)-th target edge is represented by \([l_{i,l}, r_{i,r}]\), where \( l \) is the launch node and \( r \) is the recovery node. Each target edge assigned to the drone has a set of launch and recovery node pairs \( LR_i = \{ [l_{i,1}, r_{i,1}], [l_{i,2}, r_{i,2}], \ldots \} \) that satisfy the drone battery capacity constraint, where each pair of nodes \( l \) and \( r \) should meet constraint (16–18) in Section 3. The target edges accessed by a drone flight have the same set of launch and recovery node pairs.

The variable \( A_i \), used to store the target edge information, is a matrix variable of size \( N \times 6 \), where \( N \) is the number of target edges. Each row of this matrix stores one target
edge information. The first column stores visitors of the target edges, where -1 means accessed by the vehicle, and 1 by drones. The number \( i \) in the second column means that the target edge belongs to the set \( SVTE_i \). The third column represents the order in which the target edges are accessed, labeled 1 to \( N \). The access order of target edges in the same set is contiguous. The fourth column stores the access direction of the target edges, wherein 0 indicates the forward direction that the target edge \( e_{ij} \) is accessed from \( i \) to \( j \), and 1 indicates the reverse direction. The fifth and sixth columns store the start and end nodes accessing the target edges. If the target edge is accessed by the vehicle (marked -1 in the first column), then a pair of launch and recovery nodes accessing these target edges is randomly generated. Then, Algorithm 1, which gives the “Drone-First, Vehicle-Then” heuristic method, is designed to generate the initial vehicle and drone routes.

The heuristic method first initializes the vehicle route, drone route, and the number of drone routes \( f \) (line 1), and obtains the set of vehicle target edges \( TV \) and the set of drone target edges \( TD \). It is worth noting that \( TV \) and \( TD \) here are the sets of target edges that have been determined and assigned to the vehicle and the drones, respectively. In the main loop of generating the initial solution (lines 4–13), the drone routes \( \sum_f S^{dr,f} \) are generated first (lines 4–8), and then the vehicle route \( S^{veh} \) is generated, until all target edges are assigned.

The process of generating a drone route \( S^{dr,f} \) can be described as: for the target edge \( i \) assigned to the drones, randomly select a pair of launch and recovery nodes in \( LR_i \), and then determine the drone route according to the direction and access order of all target edges in the same set as \( i \). Finally, delete the target edges from \( TD \). Except for accessing the target edges, the drones fly along the shortest distance between any two nodes in the road network, and the distance between the two nodes passed by a drone is the Euclidean distance [46].

The process of generating the vehicle route \( S^{veh} \) can be described as follows: first, the launch and recovery nodes of each drone route are continuously inserted into the vehicle route, according to the access order. Next, for the target edge \( i \) assigned to the vehicle, two endpoints are inserted into adjacent positions in the vehicle route, according to the direction and access order. Finally, the target edge \( i \) is removed from the set \( TV \). The vehicle

| Target edge | Access by which | SVTE. | Access order | Direction | Start and end nodes |
|-------------|----------------|-------|--------------|-----------|--------------------|
| Target edge 1 | 1 | 1 | 3 | 1 | 7 5 |
| Target edge 2 | -1 | 1 | 2 | 0 | 4 15 |
| Target edge 3 | 1 | 1 | 2 | 0 | 7 5 |
| Target edge 4 | 1 | 3 | 5 | 1 | 9 2 |
| Target edge 5 | 1 | 3 | 6 | 1 | 9 2 |
| Target edge 6 | -1 | 4 | 4 | 0 | 22 13 |

Figure 3. Encoding of matrix variable \( A \).

4.2. Initial Solution Generation

First, the target edge information in the matrix variable \( A \), and the launch and recovery node pairs \( LR_i = \{[l_i,1, r_{i,1}], [l_i,2, r_{i,2}], \ldots \} \) of the target edges assigned to drones, are randomly generated. Then, Algorithm 1, which gives the “Drone-First, Vehicle-Then” heuristic method, is designed to generate the initial vehicle and drone routes.

The heuristic method first initializes the vehicle route, drone route, and the number of drone routes \( f \) (line 1), and obtains the set of vehicle target edges \( TV \) and the set of drone target edges \( TD \). It is worth noting that \( TV \) and \( TD \) here are the sets of target edges that have been determined and assigned to the vehicle and the drones, respectively. In the main loop of generating the initial solution (lines 4–13), the drone routes \( \sum_f S^{dr,f} \) are generated first (lines 4–8), and then the vehicle route \( S^{veh} \) is generated, until all target edges are assigned.

The process of generating a drone route \( S^{dr,f} \) can be described as: for the target edge \( i \) assigned to the drones, randomly select a pair of launch and recovery nodes in \( LR_i \), and then determine the drone route according to the direction and access order of all target edges in the same set as \( i \). Finally, delete the target edges from \( TD \). Except for accessing the target edges, the drones fly along the shortest distance between any two nodes in the road network, and the distance between the two nodes passed by a drone is the Euclidean distance [46].

The process of generating the vehicle route \( S^{veh} \) can be described as follows: first, the launch and recovery nodes of each drone route are continuously inserted into the vehicle route, according to the access order. Next, for the target edge \( i \) assigned to the vehicle, two endpoints are inserted into adjacent positions in the vehicle route, according to the direction and access order. Finally, the target edge \( i \) is removed from the set \( TV \). The vehicle
can only travel along the road network, so the shortest distance between any two nodes in the vehicle route is calculated by Dijkstra algorithm [47].

\begin{algorithm}[h]
\caption{“Drone-First, Vehicle-Then” heuristic method}
\textbf{Input:} The target edge information $A$; launch and recovery node pairs $LR_i$
\textbf{Output:} Initial solution $S$

\begin{algorithmic}[1]
\State \textbf{Initialize} vehicle route $S_{veh} \leftarrow []; \text{ the } f\text{-th drone route } S_{dr,f} \leftarrow []; f \leftarrow 0$
\State \textbf{GET TV and TD from } $A$
\While {$TV \cup TD \neq \varnothing$}
\hspace{1em} \If {$i \in TD$}
\hspace{2em} $f \leftarrow f + 1$
\hspace{2em} Randomly select a pair of launch and recovery nodes, and determine a drone route $S_{dr,f}$ according to the direction and access order of each target edge in the same set as $i$
\hspace{2em} Delete all target edges that are in the same set as target edge $i$ from $TD$
\hspace{2em} Insert the launch and recovery nodes of target edge $i$ into the vehicle route $S_{veh}$ according to the access order
\hspace{1em} \Elsif {$i \in TV$}
\hspace{2em} Insert the two endpoints of target edge $i$ into the vehicle route $S_{veh}$ according to the access order
\hspace{2em} Delete target edge $i$ from $TV$
\hspace{1em} \Elsif
\hspace{2em} \Endif
\Endwhile
\State $S = S_{veh} \cup \sum_f S_{dr,f}$
\end{algorithmic}
\end{algorithm}

4.3. Framework of IALNS Algorithm

IALNS takes a large neighborhood search [48] as the basic search framework, and integrates a variety of effective mechanisms, such as adaptive neighborhood selection mechanism, tabu table, simulated annealing mechanism, initial solution generation method, and neighborhood search strategies, to improve the performance of IALNS. The objective of IALNS is to minimize the total patrol time. Algorithm 2 describes the framework of IALNS in detail.

IALNS inputs the target edge information $A$, initial temperature $T_0$, final temperature $T_{min}$, maximum number of iterations $I_{max}$, annealing rate $\gamma$, destroy operator set $DO$, and repair operator set $RO$, and outputs the optimal solution $S_{best}$. First, the initial solution $S$ is obtained by the “Drone-First, Vehicle-Then” heuristic method introduced in Section 4.2. Then, the weight of the destroy operators $do$, weight of the repair operators $ro$, as well as current iteration $I$, current temperature $T$, and optimal solution $S_{best}$ are initialized in line 2. In the main loop of IALNS, the solution is iteratively updated until the termination criterion is satisfied (lines 3–19). The destroy operator and the repair operator are selected and executed to obtain a new solution $S'$ (line 4–5), and the new solution is accepted according to the simulated annealing mechanism. If the optimal solution is worse than the new solution, accept the new solution $S'$; otherwise, accept $S'$ with a probability $\exp(\Delta f/T) > \zeta$, where $\zeta$ is randomly sampled in the interval $[0, 1]$ (line 7–14). Finally, at the end of each main loop, update $do$ and $ro$, the tabu table, the current iteration $I$, and current temperature $T$ accordingly (lines 15–18).
Algorithm 2: Framework of IALNS

```
Input: A; T0; Tmin; lmax; γ; DO; RO
Output: Sbest
1 S ← DFVT(A, SEj)
2 Initialize do = (1, ... , 1); ro = (1, ... , 1); I ← 1; T ← T0; Sbest ← S
3 While T > Tmin and I < lmax do
   4 Select the destroy operator d ∈ DO and repair operator r ∈ RO according to do and ro
   5 S′ = r(d(S))
   6 △f = F(S′) − F(S)
   7 If accept S′ then
      8 S ← S′
      9 If F(S) < F(Sbest) then
         10 Sbest ← S′
      Else if exp(△f/T) > ζ then
         12 Sbest ← S′
      End if
   13 End if
   14 Update do and ro
   15 Update the tabu table
   16 I ← I + 1
   17 T ← γT
4.4. Neighborhood Search Strategies

The neighborhood search strategies designed in this paper are composed of destroy operators and repair operators in pairs. The destroy operators delete the target edges from the original vehicle route and perform certain operations on the target edges, while the repair operators reinsert the operated target edges into the vehicle route with a certain rule to update the vehicle route.

Since the encoding of target edges in the matrix variable A is randomly generated, changing these elements may improve the solution in the neighborhood search. According to the characteristics of the proposed DV-ARP, we design a total of 7 destroy operators, which are described as follows:

4.4.1. Change the Access Direction of Target Edges Assigned to Drones

Randomly select a target edge assigned to drones and change the access direction of target edges, as Figure 4 shows. If there is only one target edge in the drone route, the access direction of the target edge is changed, and the start and end nodes are exchanged, and then deleted from the vehicle route. If there are multiple target edges, the entire drone route is flipped, and the start and end nodes are deleted from the vehicle route.
Figure 4. Change the access direction of target edges assigned to drones. (a) For one target edge in the drone route; (b) for multiple target edges. Different numbers in the figure represent different nodes of the road network.

4.4.2. Change the Access Direction of a Target Edge Assigned to the Vehicle

Select a target edge accessed by the vehicle randomly, change the access direction, and delete the endpoints of the target edge from the vehicle route, as shown in Figure 5.

Figure 5. Change the access direction of a target edge assigned to the vehicle. Different numbers in the figure represent different nodes of the road network.

4.4.3. Change the Start and End Nodes of a Drone Route

A target edge accessed by drones is randomly selected and changed, as Figure 6 displays. Randomly re-select a pair of launch and recovery nodes in $LR_i$, and delete the original launch and recovery nodes from the vehicle route.

Figure 6. Change the start and end nodes of a drone route. Different numbers in the figure represent different nodes of the road network.

4.4.4. Delete Target Edge

Randomly select a target edge, if it is accessed by a drone, delete the start and end nodes from the vehicle route, as in Figure 7a. If it is accessed by the vehicle, the endpoints of the target edge are directly deleted from the vehicle route, as in Figure 7b.
Reorganize two drone routes. Different numbers in the figure represent different nodes of the road network.

4.4.5. Merge Two Drone Routes

Randomly choose two drone routes, and merge the target edges in the two routes to form one drone route. Then, the original two pairs of start and end nodes are deleted from the vehicle route, which is depicted in Figure 8.

Figure 8. Merge two drone routes. Different numbers in the figure represent different nodes of the road network.

4.4.6. Reorganize Two Drone Routes

Randomly choose two drone routes and select a target edge from each route separately. Put these two target edges in one set, and put the remaining target edges of the two drone routes in another set. The start and end nodes of the two sets are randomly generated, and the two drone routes are formed accordingly. Delete the original start and end nodes from the vehicle route. The operator is presented in Figure 9.

Figure 9. Reorganize two drone routes. Different numbers in the figure represent different nodes of the road network.

4.4.7. Change the Assignment Object of the Target Edge

First, randomly select a target edge. If it is accessed by a drone, determine whether the target edge can be assigned to a vehicle. If so, assign the target edge to the vehicle, and
delete the start and end nodes from the vehicle, as shown in Figure 10a. If it is accessed by the vehicle, randomly generate a pair of start and end nodes of the target edge to form a new drone route, and the two endpoints are deleted from the vehicle route. In Figure 10b, the target edge $e_{78}$ originally assigned to the vehicle is reassigned to the drones.

![Figure 10. Change the assignment object of the target edge.](image)

(a) Drone to vehicle; (b) vehicle to drone. Different numbers in the figure represent different nodes of the road network.

4.4.8. The Repair Operator

The repair operator can be abstracted, as in Figure 11. The two endpoints of the target edge accessed by the vehicle, or the start and end nodes of the target edge accessed by the drone after the destroy operator operation, are randomly inserted into the vehicle route. Note that the two endpoints of the target edge assigned to the vehicle must be inserted into adjacent positions.

![Figure 11. The repair operator.](image)

In any destroy operator operations, if the number of drone flights increases, it is necessary to determine whether the number of drones performing tasks during the interval of a newly added drone flight is less than the number of drones carried by the vehicle. If not, skip this operation to ensure that the constraints (11–13) in Section 3 are satisfied.

The combination of the above seven destroy operators and the repair operator constitutes seven different neighborhood search strategies.

5. Experiments and Discussion

In this section, we first set up the experimental parameters, designed the instances of different scales, and introduced the comparison algorithms. Next, we tested the performance of the proposed algorithm through simulation experiments. Finally, a practical case was applied, to further verify the applicability of the algorithm. All algorithms were implemented in Python, and run on an x64 Windows 10 computer, with I7-10700F CPU and 16.00 GB memory.
5.1. Simulation Setup

According to previous experiments, the parameters involved were configured as follows: for each instance, the maximum number of iterations $I_{\text{max}} = 1500$. The initial temperature $T_0$ was set to 100, and the termination temperature $T_{\text{min}} = 0.1$. The annealing rate $\gamma$ is 0.99. In addition, the relevant parameters of the vehicle and drones were set according to typical situations in practical applications, as shown in Table 3.

Table 3. Experimental parameter design.

| Parameters                      | Value (Unit) |
|---------------------------------|--------------|
| Number of drones                | 3            |
| Vehicle speed                   | 30 km/h      |
| Drone speed                     | 35 km/h      |
| Battery life of a drone         | 0.67 h       |

Considering the characteristics of the urban road network, an undirected network graph with 50 nodes and 79 edges was randomly generated within the range of $5 \text{ km} \times 5 \text{ km}$. Three groups of instances with 5, 10, and 20 target edges were designed, marked as E1/E2/E3, and different target edges (distinguished by A, B, and C) were randomly set for each group of instances. For each instance, the number of target edges that can only be accessed by the drones is $1/5$ of the total number of target edges. Table 4 describes the instances in detail.

Table 4. Introduction of the instances.

| No. | Instances | Total Num. of Target Edges | Num. of Target Edges Only for Drones |
|-----|-----------|---------------------------|-------------------------------------|
| 1   | E1-A      | 5                         | 1                                   |
| 2   | E1-B      | 5                         | 1                                   |
| 3   | E1-C      | 5                         | 1                                   |
| 4   | E2-A      | 10                        | 2                                   |
| 5   | E2-B      | 10                        | 2                                   |
| 6   | E2-C      | 10                        | 2                                   |
| 7   | E3-A      | 20                        | 4                                   |
| 8   | E3-B      | 20                        | 4                                   |
| 9   | E3-C      | 20                        | 4                                   |

In this paper, the proposed IALNS was compared with VND [49], VND with tabu list (VND-tabu), and the improved LNS [48] (ILNS). Compared with IALNS, the neighborhood search strategies were randomly selected with no weight change in ILNS. Moreover, the length of the table list is set as follows:

$$L_{\text{max}} = 2 \left\lfloor \sqrt{\frac{N(N-1)}{2}} \right\rfloor$$  \hspace{1cm} (19)

where $N$ represents the total number of target edges.

5.2. Experiments and Analyses

All comparison algorithms used the same neighborhood search strategies and initial solution generation method as IALNS, for fairness. We ran each algorithm 20 times for each instance. Table 5 and Figure 12 present the average performance of IALNS and the three comparison algorithms. “Result (s)” and “Running time (s)” represent the total patrol time and consumed running time, respectively. “Average” is the average value of the nine
instances, and “GAP” is a parameter used to further compare the performance differences between IALNS and the other approaches, which can be calculated by

$$\text{GAP}(X, \text{IALNS}) = \frac{f_X - f_{\text{IALNS}}}{\max(f_X, f_{\text{IALNS}})} \times 100\%$$ \hspace{1cm} (20)

Table 5. The results and running time of each instance on IALNS and the other three comparison algorithms.

| Instances | VND Result (s) | VND Running Time (s) | VND-Tabu Result (s) | VND-Tabu Running Time (s) | ILNS Result (s) | ILNS Running Time (s) | IALNS Result (s) | IALNS Running Time (s) |
|-----------|---------------|----------------------|---------------------|--------------------------|----------------|----------------------|----------------|----------------------|
| E1-A      | 2521          | 27.85                | 2452                | 27.42                    | 2430          | 23.96                | 2376           | 27.92                |
| E1-B      | 2751          | 21.18                | 2878                | 25.56                    | 2923          | 11.42                | 2647           | 22.18                |
| E1-C      | 2542          | 25.02                | 2521                | 25.93                    | 2570          | 21.29                | 2647           | 27.32                |
| E2-A      | 3108          | 40.16                | 3212                | 43.25                    | 3086          | 37.56                | 2960           | 36.30                |
| E2-B      | 3163          | 46.78                | 3153                | 47.77                    | 3092          | 43.49                | 3013           | 43.78                |
| E2-C      | 3079          | 38.20                | 3098                | 40.86                    | 3054          | 37.47                | 3015           | 36.18                |
| E3-A      | 5630          | 106.58               | 5432                | 104.03                   | 5345          | 92.92                | 5108           | 90.36                |
| E3-B      | 5167          | 120.09               | 5398                | 127.58                   | 5081          | 105.68               | 4986           | 105.48               |
| E3-C      | 5671          | 90.25                | 5817                | 96.22                    | 5027          | 82.86                | 4855           | 73.58                |

Average: 3737 - 3773 - 3623 - 3492 -  
GAP (%): 6.56% - 7.45% - 3.62% - 0 -

Figure 12. Comparison of results of different algorithms.

In Table 5 and Figure 12, it can be observed that as the number of target edges increases, the solution results and running time of each algorithm increase. From Table 5, the comparison of all algorithms shows the superiority of IALNS. Compared with the three comparison algorithms, IALNS reduces the total patrol time by 6.56%, 7.45%, and 3.62% respectively. As Figure 12 displays, ILNS and IALNS always perform better solutions. It is because ILNS and IALNS have a larger search space, by randomly selecting the neighborhood search strategies, which increases the possibility of finding better solutions. Compared with ILNS and IALNS, IALNS has obvious advantages in the solution results, which can be explained as the adaptive selection mechanism dynamically adjusting the weight of each neighborhood search strategy to guide the search direction, in order to obtain better solutions. Moreover, IALNS performs best in terms of solution quality and running speed. In general, the IALNS algorithm combines the advantages of the adaptive mechanism, simulated annealing mechanism, and tabu table, which not only avoids falling into local optimum, but also improves the convergence speed.
Taking the three instances E1-A, E2-A, and E3-A as representatives, the schematic diagrams of the route planning results are shown in Figure 13. The vehicle routes are represented by solid black lines, and the routes of different drones are labeled in different colors. The forward directions of all routes are indicated by arrows. The blue dots label the depot. Target edges that can be accessed by the vehicle or drones are represented by red segments, and target edges that can only be accessed by drones are represented by pink segments.

Figure 13. Schematic diagrams of route planning results on different instances with different number of target edges. (a) E1-A; (b) E2-A; (c) E3-A.

To further verify the stability of each algorithm, the statistical results of the standard deviation of different algorithms are obtained, as shown in Table 6. Each algorithm was independently run 20 times. “Max”, “Min” and “Avg” are the maximum, minimum, and average values of the standard deviations of the nine instances, respectively.

Table 6. The statistical results of the standard deviation of different algorithms.

| Instances | Total Num. of Target Edges | Algorithms  | VND | VND-Tabu | ILNS | IALNS |
|-----------|-----------------|--------------|-----|----------|------|-------|
| E1-A      | 5               | VND          | 0.0829 | 0.0777 | 0.0681 | 0.0770 |
| E1-B      | 5               | VND-Tabu     | 0.1341 | 0.0496 | 0.0519 | 0.1312 |
| E1-C      | 5               | ILNS         | 0.0660 | 0.0551 | 0.0638 | 0.0580 |
| E2-A      | 10              | IALNS        | 0.1399 | 0.1620 | 0.0953 | 0.0710 |
| E2-B      | 10              | VND          | 0.1195 | 0.0560 | 0.0511 | 0.0514 |
| E2-C      | 10              | VND-Tabu     | 0.0690 | 0.0918 | 0.0542 | 0.0849 |
| E3-A      | 20              | ILNS         | 0.1329 | 0.1663 | 0.0902 | 0.1262 |
| E3-B      | 20              | IALNS        | 0.1938 | 0.1945 | 0.2235 | 0.0792 |
| E3-C      | 20              | VND          | 0.2432 | 0.2603 | 0.2003 | 0.0643 |
| Max       |                 |              | 0.2432 | 0.2603 | 0.2235 | 0.1312 |
| Min       |                 |              | 0.0660 | 0.0496 | 0.0511 | 0.0514 |
| Avg       |                 |              | 0.1313 | 0.1237 | 0.0998 | 0.0826 |

The statistical results in Table 6 show that the average standard deviation of IALNS is 0.0826, which is smaller than that of the other comparison algorithms. The standard deviation of the VND-tabu algorithm performs better than that of VND, due to the tabu table, which reduces the repeated search of the same target edge in a short term. The standard deviation of IALNS is smaller than that of ILNS, indicating that the adaptive selection mechanism of the neighborhood search strategies improves the stability of the algorithm, to a certain extent.

In order to verify the impact of the number of drones on the experimental results, a sensitivity analysis based on the three instances, E1_A, E2_A, and E3_A is designed. For each instance, the number of drones is set from one to nine. Each algorithm is run 10 times repeatedly. All experimental results are recorded in Figure 14 and Table 7.
As depicted in Figure 14 and Table 7, in the three instances, as the number of drones increases, the experimental results gradually decrease, and the IALNS algorithm still has obvious advantages. In particular, the reduction is greatest when the number of drones increases from one to three. However, when the number of drones increases from three to nine, the improvement effect on the experimental results decreases with each additional drone. When the number of drones increases by a certain value, the drone resources are saturated, and the drones are enough to meet the patrol needs. At this time, even if the number of drones carried by the vehicle increases, the increased drones will not be dispatched for patrolling. As Figure 14 displays, for instances E1_A and E2_A, three drones can saturate, while E3_A may need six drones (although the gain from increasing the drones from three to six is small). The number of drones currently taken matches the actual demand for traffic patrols. Assigning more drones to the vehicle might save patrol time, but the improvement in cost is huge. Therefore, the choice of the number of drones shows a trade-off between patrol time and cost.

When the approach proposed in this paper is applied to the smart city context, the speed of the vehicle may be significantly improved, and the experimental results may vary greatly. Therefore, we also take the instances E1_A, E2_A, and E3_A as examples to design analysis experiments on the speed of the vehicle. The vehicle speed is set at 30–60 km/h, with an interval of 5 km/h. Each algorithm is run 10 times independently. Figure 15 and Table 8 show the experimental results at different speeds of the vehicle.
As depicted in Figure 14 and Table 7, in the three instances, as the number of drones increases from one to three, the experimental results decrease. However, when the number of drones increases from three to six, the experimental results remain almost constant. Therefore, we also take the instances E1_A, E2_A, and E3_A as examples to study the effects of vehicle speed on experimental results. The effects of the speed of the vehicle on experimental results are shown in Figure 15. The experimental results decrease significantly as the vehicle speed increases, and IALNS always performs best.

| Instances | Algorithms | Speed of the Vehicle |
|-----------|------------|----------------------|
| E1_A      | VND        | 30 2520 35 2382 40 2284 45 2203 50 2138 55 1938 60 1826 |
|           | VND-tabu   | 30 2497 35 2351 40 2314 45 2292 50 2165 55 1981 60 1923 |
|           | ILNS       | 30 2571 35 2434 40 2370 45 2287 50 2244 55 2188 60 2075 |
|           | IALNS      | 30 2458 35 2331 40 2153 45 2080 50 2041 55 1914 60 1754 |
| E2_A      | VND        | 30 3090 35 2993 40 2779 45 2547 50 2424 55 2301 60 2417 |
|           | VND-tabu   | 30 3119 35 2980 40 2767 45 2654 50 2526 55 2457 60 2572 |
|           | ILNS       | 30 3145 35 3033 40 2882 45 2708 50 2554 55 2486 60 2503 |
|           | IALNS      | 30 2993 35 2862 40 2658 45 2531 50 2498 55 2421 60 2409 |
| E3-A      | VND        | 30 5733 35 4967 40 4902 45 4596 50 4401 55 3960 60 3778 |
|           | VND-tabu   | 30 5585 35 5187 40 5039 45 4527 50 4279 55 3965 60 3759 |
|           | ILNS       | 30 5423 35 4981 40 4826 45 4608 50 4297 55 3963 60 4042 |
|           | IALNS      | 30 4853 35 4786 40 4508 45 4404 50 4076 55 3861 60 3582 |

The effects of the speed of drones on the performances of the algorithms are obvious. As Figure 15 and Table 8 display, in the three instances, the experimental results decrease significantly as the vehicle speed increases, and IALNS always performs best. It follows that the increased speed of the vehicle is an effective consideration to reduce the patrol time, which puts forward higher requirements for the state of the urban traffic network. However, when the vehicle speed is doubled from 30 km/h to 60 km/h, the total patrol time is reduced by less than half. This is because, despite the increase in vehicle speed, the unchanged speed of drones, and the potential increase in waiting time, make the reduction in patrol time less than desirable.

5.3. An Actual Case Study

In order to further study the application significance of the VD-ARP proposed in this paper, an actual case in the real world is considered. The simplified road network in a certain area of Changsha City is shown in Figure 16. There are 56 intersections and 94 road sections in this area, which are represented by small circles and line segments, respectively. The eight road sections that can be patrolled by vehicles or drones, and two road sections that can only be patrolled by drones are randomly set, marked with red and pink lines, respectively.
The parameter settings are the same as in Section 5.1. The patrol time (second) with different numbers of drones is presented in Table 9, and the route planning of the vehicle and the drones, obtained by IALNS, is shown in Figure 17.

Table 9. Patrol time (s) with different numbers of drones.

| Algorithms       | Num. of Drones |
|------------------|---------------|
|                  | 1  | 2  | 3  | 4  | 5  | 6  |
| VND              | 3352| 2777| 2386| 2292| 2384| 2441|
| VND-tabu         | 3268| 2516| 2353| 2202| 2198| 2347|
| ILNS             | 3146| 2391| 2371| 2211| 2078| 2294|
| IALNS            | 2980| 2265| 2283| 2109| 2002| 2184|

Figure 16. Simplified road network in a certain area of Changsha City.

Figure 17. Route planning of the vehicle and the drones.

In this case study, by setting appropriate parameters, such as the vehicle speed, drone speed, and the number of drones, a feasible route planning scheme for the vehicle and
drones is obtained within about 40 s of running time, which meets the needs of urban traffic patrols. There are 10 road sections to be patrolled, with a total length of 14.56 km. The vehicle routes are represented by solid black lines, and the routes of different drones are labeled in different colors. It can be seen from Table 9 that when the number of drones is three, the patrol time obtained by the IALNS algorithm is 2283 s, that is, 0.63 h. Hence, a satisfactory solution is obtained within an acceptable time. Compared with the other three comparison algorithms, IALNS has obvious advantages. It is worth noting that in the route planning scheme given in Figure 17, there are at most two drones patrolling at the same time, that is, at least two drones are needed. The additional drones are barely effective. Through the coordination of multiple drones and a vehicle, the efficiency of urban traffic patrols is significantly improved.

6. Conclusions

Proper traffic patrols improve the safety of urban traffic. To improve the efficiency and flexibility of traffic patrol, a coordinated vehicle–drone arc routing problem, with a vehicle and multiple drones (DV-ARP), is proposed for planning the parallel routes of a vehicle and drones in urban traffic patrols. To solve the problem, an improved, adaptive, large neighborhood search algorithm (IALNS) is designed. Firstly, the initial solution is generated by the heuristic method of “Drone-First, Vehicle-Then”. Secondly, a set of destroy operators and a repair operator are designed to generate new candidate solutions. Finally, the simulated annealing mechanism, tabu table, and multiple neighborhood search strategies are integrated into the framework of an adaptive large neighborhood search algorithm, in order to search the solution space and find the optimal solution.

Experiments with different dimensions are conducted to explore the effectiveness of IALNS. The superiority of IALNS is verified by numerical experiments compared with other algorithms on different scales. The results show that the proposed approach IALNS reduces the total patrol time by at most 7.45%, compared with the three comparison algorithms (i.e., VND, VND-tabu, and ILNS). Additionally, the performance of all the four algorithms shows that all of them address the instances within different scales. To further test the impact of the main parameters on the results, two critical factors are evaluated, including the drone number carried by the vehicle and the vehicle speed. As the number of drones increases from one to three, the total patrol time has a dramatic decline, then slows down, and finally stops decreasing. It means that patrol scenarios with different numbers of target edges have different requirements for the drones. In addition, the total patrol time also decreases as the vehicle speed increases from 30 to 60 km/h. A real-world road network in Changsha City is simulated to further verify the applicability of IALNS.

The proposed IALNS algorithm in this paper is expected to provide effective route planning schemes for traffic management departments to carry out urban traffic patrols or collect traffic data. However, it ignores some complex constraints that may need to be considered in practical applications, such as energy consumption and separated launching/recovering time of drones, as well as dynamic demand, which would be further studied in the future. In addition, how to design a reasonable evaluation system to explore the balance between drone costs and patrol efficiency is also worth studying.

Author Contributions: Conceptualization, G.W. and K.Z.; methodology, K.Z.; software, J.C.; validation, G.W., K.Z. and J.C.; formal analysis, J.C. and M.M.; investigation, K.Z.; resources, G.W.; data curation, K.Z. and J.C.; writing—original draft preparation, G.W.; writing—review and editing, K.Z. and J.C.; visualization, K.Z.; supervision, M.M.; project administration, G.W.; funding acquisition, G.W. All authors have read and agreed to the published version of the manuscript.

Funding: This study is fully supported by the National Natural Science Foundation of China, grant number 62073341, and the Natural Science Fund for Distinguished Young Scholars of Hunan Province (Grant No. 2019JJ20026).

Institutional Review Board Statement: Not applicable.

Informed Consent Statement: Not applicable.
Data Availability Statement: Not applicable.

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

References
1. Lou, Y.; Yin, Y.; Lawphongpanich, S. Freeway service patrol deployment planning for incident management and congestion mitigation. Transp. Res. Part C Emerg. Technol. 2011, 19, 283–295. [CrossRef]
2. Besada, J.A.; Carramiñana, D.; Bergesio, L.; Campaña, I.; Bernardos, A.M. Modelling and Simulation of Collaborative Surveillance for Unmanned Traffic Management. Sensors 2022, 22, 1498. [CrossRef][PubMed]
3. Doliri, K.; Heinrichs, J.; Messier, G.G.; Magierowski, S. Vehicle routing problems for drone delivery. IEEE Trans. Syst. Man Cybern. Syst. 2017, 47, 70–85. [CrossRef]
4. Murray, C.C.; Chu, A.G. The flying sidekick traveling salesman problem: Optimization of drone-assisted parcel delivery. Transp. Res. Part C Emerg. Technol. 2015, 54, 86–109. [CrossRef]
5. Chung, S.H.; Sah, B.; Lee, J. Optimization for drone and drone-truck combined operations: A review of the state of the art and future directions. Comput. Oper. Res. 2020, 120, 105004. [CrossRef]
6. Agatz, N.; Bouman, P.; Schmidt, M. Optimization approaches for the traveling salesman problem with drone. Transp. Sci. 2018, 52, 965–981. [CrossRef]
7. Tu, P.A.; Dat, N.T.; Dung, P.Q. Traveling salesman problem with multiple drones. In Proceedings of the Ninth International Symposium on Information and Communication Technology, Danang, Vietnam, 6–7 December 2018; pp. 46–53.
8. Han, Y.-q.; Li, J.-q.; Liu, Z.; Liu, C.; Tian, J. Metaheuristic algorithm for solving the multi-objective vehicle routing problem with time window and drones. Int. J. Adv. Robot. Syst. 2020, 17, 1–14. [CrossRef]
9. 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 Trans. Intell. Transp. Syst. 2021, 1–15. [CrossRef]
10. Hu, M.; Liu, W.; Lu, J.; Fu, R.; Peng, K.; Ma, X.; Li, J. On the joint design of routing and scheduling for Vehicle-Assisted Multi-UAV inspection. Future Gener. Comput. Syst. 2019, 94, 214–223. [CrossRef]
11. Karak, A.; Abdelghany, K. The hybrid vehicle-drone routing problem for pick-up and delivery services. Transp. Res. Part C Emerg. Technol. 2019, 102, 427–449. [CrossRef]
12. Poikonen, S.; Golden, B. Multi-visit drone routing problem. Comput. Oper. Res. 2020, 113, 104802. [CrossRef]
13. Liu, Y.; Liu, Z.; Shi, J.; Wu, G.; Pedrycz, W. Two-echelon routing problem for parcel delivery by cooperated truck and drone. IEEE Trans. Syst. Man Cybern. Syst. 2020, 51, 7450–7465. [CrossRef]
14. Peng, K.; Du, J.; Lu, F.; Sun, Q.; Dong, Y.; Zhou, P.; Hu, M. A hybrid genetic algorithm on routing and scheduling for vehicle-assisted multi-drone parcel delivery. IEEE Access 2019, 7, 49191–49200. [CrossRef]
15. de Freitas, J.C.; Penna, P.H.V. A variable neighborhood search for flying sidekick traveling salesman problem. Int. Trans. Oper. Res. 2020, 27, 267–290. [CrossRef]
16. Es Yurek, E.; Ozmutlu, H.C. A decomposition-based iterative optimization algorithm for traveling salesman problem with drone. Transp. Res. Part C Emerg. Technol. 2018, 91, 249–262. [CrossRef]
17. Corberan, A.; Eglese, R.; Hasle, G.; Plan, I.; Sanchis, J.M. Arc routing problems: A review of the past, present, and future. Networks 2021, 77, 88–115. [CrossRef]
18. Macrina, G.; Pugliese, L.D.; Guerriero, F.; Laporte, G. Drone-aided routing: A literature review. Transp. Res. Part C Emerg. Technol. 2020, 120, 25. [CrossRef]
19. Liu, Y.; Shi, J.; Liu, Z.; Huang, J.; Zhou, T. Two-layer routing for high-voltage powerline inspection by cooperated ground vehicle and drone. Energies 2019, 12, 1385. [CrossRef]
20. Luo, H.; Zhang, P.; Wang, J.; Wang, G.; Meng, F. Traffic patrolling routing problem with drones in an urban road system. Sensors 2019, 19, 5164. [CrossRef]
21. Babić, D.; Babić, D.; Fiolić, M.; Eichberger, A.; Magosi, Z.F. Impact of Road Marking Retroreflectivity on Machine Vision in Dry Conditions: On-Road Test. Sensors 2022, 22, 1303. [CrossRef]
22. Steil, D.A.; Pate, J.R.; Kraft, N.A.; Smith, R.K.; Dixon, B.; Ding, L.; Parrish, A. Patrol routing expression, execution, evaluation, and engagement. IEEE Trans. Intell. Transp. Syst. 2010, 12, 58–72. [CrossRef]
23. Jalili, M.; Ozbek, M.E.; McCarthy, J.G.; Rabbani, E.R.K. Effective Performance Measures for Highway Patrol Agencies to Change Poor Driver Behaviors. Procedia Manuf. 2015, 3, 2450–2457. [CrossRef]
24. Li, P.; Walton, J.R. Evaluating freeway service patrols in low-traffic areas using discrete-event simulation. J. Transp. Eng. 2013, 139, 1059–1104. [CrossRef]
25. Chawathe, S.S. Organizing hot-spot police patrol routes. In Proceedings of the 2007 IEEE Intelligence and Security Informatics, New Brunswick, NJ, USA, 23–24 May 2007; pp. 79–86.
26. Wang, Z.; He, W.; Zhang, X.; Wang, Y.; Wu, B.; Wang, Y. Lane-based vehicular speed characteristics analysis for freeway work zones using aerial videos. Can. J. Civ. Eng. 2021, 48, 274–283. [CrossRef]
27. Munishkin, A.A.; Milutinovi, D.; Casbeer, D.W. Minmax time efficient inspection of ground vehicles by a UAV team. Robot. Auton. Syst. 2020, 125, 103370. [CrossRef]
28. Wu, D.; Arkhipov, D.I.; Kim, M.; Talcott, C.L.; Regan, A.C.; McCann, J.A.; Venkatasubramanian, N. ADDSEN: Adaptive data processing and dissemination for drone swarms in urban sensing. IEEE Trans. Comput. 2016, 66, 183–198. [CrossRef]
29. Erdős, P. Graph Theory and Probability. Can. J. Math. 1959, 11, 34–38. [CrossRef]
30. Eiselt, H.A.; Gendreau, M.; Laporte, G. Arc routing problems, part I: The Chinese postman problem. Oper. Res. 1995, 43, 231–242. [CrossRef]
31. Nilofer, M.; Rizwanullah, M. An implementation of Chinese postman problem with priorities. J. Intell. Fuzzy Syst. 2020, 38, 2979–2989. [CrossRef]
32. Calogiuri, T.; Ghiani, G.; Guerriero, E.; Mansini, R. A branch-and-bound algorithm for the time-dependent rural postman problem. Comput. Oper. Res. 2019, 102, 150–157. [CrossRef]
33. Monroy-Licht, M.; Amaya, C.A.; Langevin, A. Adaptive large neighborhood search algorithm for the rural postman problem with time windows. Networks 2017, 70, 44–59. [CrossRef]
34. Cerrone, C.; Dussault, B.; Wang, X.; Golden, B.; Wasil, E. A two-stage solution approach for the directed rural postman problem with turn penalties. Eur. J. Oper. Res. 2019, 272, 754–765. [CrossRef]
35. Xing, L.; Rohlfshagen, P.; Chen, Y.; Yao, X. An evolutionary approach to the multidepot capacitated arc routing problem. IEEE Trans. Evol. Comput. 2009, 14, 356–374. [CrossRef]
36. Constantino, M.; Gouveia, L.; Mourao, M.C.; Nunes, A.C. The mixed capacitated arc routing problem with non-overlapping routes. Eur. J. Oper. Res. 2015, 244, 445–456. [CrossRef]
37. Kwan, M. Programming method using odd or even pints. Acta Math. Sin. 1960, 10, 263–266.
38. Lima, V.H.R.; Lima, E.D.; Sherafat, H. Routing of vehicles in the delivery/collection problems-Application of a modified Ant Colony Algorithm. Rev. Bras. Comput. Apl. 2020, 12, 44–53. [CrossRef]
39. Hertz, A.; Laporte, G.; Hugo, P.N. Improvement procedures for the undirected rural postman problem. INFORMS J. Comput. 1999, 11, 53–62. [CrossRef]
40. Fernandez, E.; Laporte, G.; Rodriguez-Pereira, J. A branch-and-cut algorithm for the multidepot rural postman problem. Transp. Sci. 2018, 52, 353–369. [CrossRef]
41. Zhang, Y.; Mei, Y.; Huang, S.; Zheng, X.; Zhang, C. A Route Clustering and Search Heuristic for Large-Scale Multidepot Capacitated Arc Routing Problem. IEEE Trans. Cybern. 2021, 1–14. [CrossRef]
42. Babaee Tirkolaee, E.; Mahdavi, I.; Seyyed Esfahani, M.M.; Weber, G.-W. A hybrid augmented ant colony optimization for the multi-trip capacitated arc routing problem under fuzzy demands for urban solid waste management. Waste Manag. Res. 2020, 38, 156–172. [CrossRef]
43. Oh, H.; Kim, S.; Tsourdos, A.; White, B.A. Coordinated road-network search route planning by a team of UAVs. Int. J. Syst. Sci. 2014, 45, 825–840. [CrossRef]
44. Campbell, J.F.; Corberán, À.; Plana, I.; Sanchis, J.M. Drone arc routing problems. Networks 2018, 72, 543–559. [CrossRef]
45. Poikonen, S.; Campbell, J.F. Future directions in drone routing research. Networks 2021, 77, 116–126. [CrossRef]
46. Danielsson, P.-E. Euclidean distance mapping. Comput. Graph. Image Processing 1980, 14, 227–248. [CrossRef]
47. Dijkstra, E.W. A note on two problems in connexion with graphs. Numer. Math. 1959, 1, 269–271. [CrossRef]
48. Pisinger, D.; Ropke, S. Large neighborhood search. In Handbook of Metaheuristics; Springer: Boston, MA, USA, 2010; pp. 399–419.
49. Mladenović, N.; Hansen, P. Variable neighborhood search. Comput. Oper. Res. 1997, 24, 1097–1100. [CrossRef]