Optimization of Multi-Agent Flying Sidekick Traveling Salesman Problem over Road Networks

Ruixiao Yang, Chuchu Fan

Department of Aeronautics and Astronautics, Massachusetts Institute of Technology
ruixiao@mit.edu, chuchu@mit.edu

Abstract

The mixed truck-drone delivery system has attracted increasing attention for its potential to optimize last-mile logistics. While the Flying Sidekick Traveling Salesman Problem (FSTSP) provides a foundation for modeling the truck-drone collaboration, it falls short of capturing real-world complexities by assuming a single truck-drone pair operating on a fully connected graph. We introduce the Multi-Agent FSTSP (MA-FSTSP), which extends FSTSP to handle multiple trucks, each carrying multiple drones operating over real road networks. Trucks must follow roads, while drones can fly directly between locations. To solve this NP-hard problem efficiently, we propose a novel three-phase algorithm that first partitions customers using a set-based distance heuristic, then computes initial truck routes via a Set TSP formulation, and finally optimizes drone deployment patterns by dynamic programming. Through extensive experiments on real-world road networks from Manhattan (1,024 nodes) and Boston (11,000 nodes), we demonstrate that our method achieves more than 30% cost reduction compared to existing approaches while scaling effectively to problems with 150 customers within a 20-minute computational time-bound.

1 Introduction

In the field of aerial robotics, drone-assisted delivery shows promise for reducing road traffic and improving last-mile delivery efficiency. The Flying Sidekick Traveling Salesman Problem (FSTSP) (Murray and Chu 2015) first modeled truck-drone cooperation, though it did not consider the drone's battery limit and simplified path-finding by limiting drones to taking actions only at customer nodes.

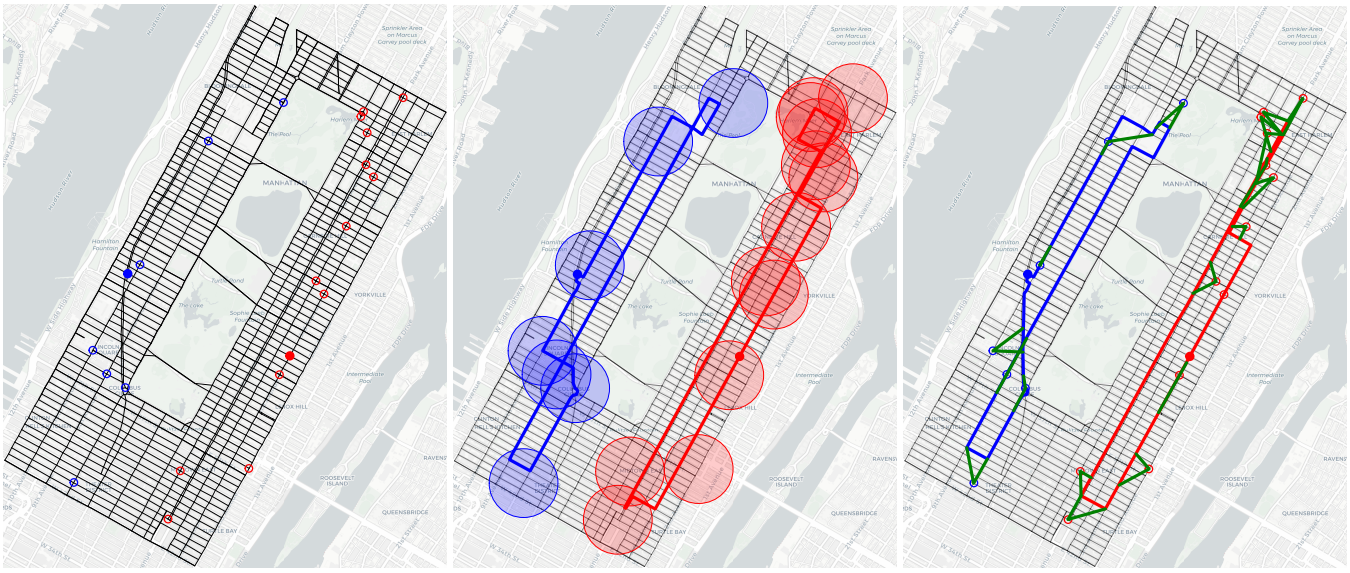
This paper introduces Multi-Agent FSTSP (MA-FSTSP), which extends FSTSP in two key aspects: (1) it considers multiple trucks, each carrying multiple drones with limited battery capacity, and (2) it relaxes the dispatch and collect location of drones to any node on the road network. In MA-FSTSP, trucks departing from different *depots* carrying drones must serve all *customers* exactly once, with drones constrained by battery capacity and payload limits. While trucks must follow the road network, drones can fly directly between locations. The objective is to minimize the total delivery cost, proportional to the truck's operating time. The special case of the MA-FSTSP with 0 drones, 1 truck, and 1 depot is the standard TSP, showing its NP-hardness.

Contribution. We develop a Mixed-Integer Linear Programming (MILP) formulation for the problem and propose a novel 3-phase method that achieves state-of-the-art performance and scalability. The first phase efficiently assigns customers to truck groups using a set-based distance heuristic. The second phase introduces Set TSP, which groups road nodes near customers to approximate feasible drone operations to determine the customers' serving sequence. The third phase performs simultaneous truck-drone route planning while maintaining the ordering of customers from Phase 2. We validate our approach through experiments on real-world road networks from Manhattan and Boston, demonstrating over 30% cost reduction compared to baselines. Our method successfully scales to large instances with 300+ customers on the Manhattan map and 150 customers on the Boston map, showing significant scalability.

Related Work. The Traveling Salesman Problem (TSP) and Vehicle Routing Problem (VRP) form the foundation of drone-assisted delivery optimization, where trucks operate from a central depot. The Flying Sidekick Traveling Salesman Problem (FSTSP) (Murray and Chu 2015) establishes the combined truck-drone delivery optimization framework.

Research has evolved along three main trajectories: single truck with multiple drones (Mbiadou Saleu et al. 2018; Karak and Abdelghany 2019; Murray and Raj 2020; Cavani, Iori, and Roberti 2021; Salama and Srinivas 2022; Bruni, Khodaparasti, and Moshref-Javadi 2022), multiple trucks each with one drone (Sacramento, Pisinger, and Ropke 2019; Chiang et al. 2019; Lu, Yang, and Yang 2022; Luo et al. 2022), and multiple trucks each with multiple drones (Poikonen, Wang, and Golden 2017; Kitjacharoenchai et al. 2019; Tamke and Buscher 2021; Chen, Demir, and Huang 2021; Gao, Zhen, and Wang 2023). Due to the computational complexity of exact solutions, researchers have developed heuristic and meta-heuristic approaches to balance solution quality and feasibility, including Clark and Wright heuristics (Karak and Abdelghany 2019), adaptive large neighborhood search (Salama and Srinivas 2022; Chen, Demir, and Huang 2021), and the column generation method (Gao, Zhen, and Wang 2023).

A key limitation of early models was their simplified rep-



(a) Phase 1: assign customers to truck groups (b) Phase 2: solve the Set-TSP to get visiting orders of customers (c) Phase 3: optimize routes for trucks and drones simultaneously based on visiting orders

Figure 1: Algorithm output for each phase. (a) Phase 1: Customer assignment, where customers (hollow circles) are assigned to their nearest depots (solid circles). (b) Phase 2: Set TSP application, showing nodes within half the drone flight distance limit (circles) and optimal tours for each truck group (colored lines). (c) Phase 3: Final optimized routes, with truck routes shown in red and blue for different groups, and drone routes in green when operating independently.

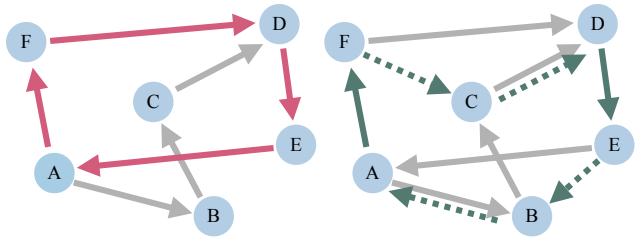
resentation of operational environments as fully connected graphs of customers and depots, and their restriction of drone actions to customer nodes only. Recent work has addressed this by incorporating more realistic synchronization settings, including allowing drones to take off and land at any point in 2D space (Carlsson and Song 2018), at any point on the arcs in the fully connected graph (Li et al. 2022), or any node in real-world road networks (Lin et al. 2022; Gao et al. 2023). Such relaxation introduces additional complexity to the problem and suffers from scalability issues.

Similar cooperation challenges also exist in Multi-Agent Path Finding (MAPF), where research on pairwise collaboration (Greshler et al. 2021) and drone-public transit coordination (Choudhury et al. 2021a,b) has yielded valuable insights for multi-agent truck-drone delivery optimization.

2 Problem Description

MA-FSTSP aims to find optimal delivery routes for multiple truck-drone groups operating on a road network, where each group consists of a delivery truck and multiple drones. The system must satisfy four key constraints:

1. Trucks must follow the road network, while drones can fly directly between any two locations;
2. Each drone can only depart from and return to its assigned truck at nodes in the network;
3. Each drone can only visit one customer per flight due to payload limitations and travel a maximum distance per flight due to battery limitations;
4. All groups must depart from and return to the same depots, serving all customers exactly once in total.



(a) Truck Route (red) (b) Drone Route (green)

Figure 2: An example of the truck and drone routes on a directed graph with node A∼F. The solid arrows (in all colors) represent the road that the truck must follow. The truck route (A, F, D, E, A) is colored red, and the drone route (A, F, C, D, E, B, A) is colored green. The solid green arrows mean the drone is carried by a truck, and the dashed green arrows mean the drone is flying freely.

Let $\mathcal{G} = (\mathcal{V}, \mathcal{E})$ be a strongly connected directed graph[1] representing the road network, where $\mathcal{V}$ is the set of vertices and $\mathcal{E}$ is the set of edges. Each pair of vertices $u, v \in \mathcal{V}$ is associated with a distance $d(u, v) \geq 0$, which may be the Euclidean distance on a 2D map or geometric distance on Earth. Let $\mathcal{P} = \{p_1, p_2, \cdots, p_m\} \subseteq \mathcal{V}$ be the set of m depots and $\mathcal{C} = \{c_1, c_2, \cdots, c_n\} \subseteq \mathcal{V}$ be the set of n customers, where $\mathcal{C} \cap \mathcal{P} = \varnothing$. We assume that depots and customers are homogeneous. Each depot $p \in \mathcal{P}$ has one truck

[1] A directed graph is called strongly connected if there is a path in each direction between each pair of graph vertices.

group $\mathcal{T}_p$ consisting of one truck moving at speed s_{tr} and k drones moving at speed s_{dr}. We use r to denote the drone's max travel distance.

We denote the plan of a truck group starting and ending at depot p by π_p. The cost of π_p, cost(π_p) is the time-based payment to the truck group with a constant unit rate, which is proportional to the completion time of the truck group. The objective is to minimize the overall cost for all truck groups.

Definition 1 (**MA-FSTSP**). *Given a strongly connected directed graph $\mathcal{G} = (\mathcal{V}, \mathcal{E})$, and m truck groups starting from different depots in set $\mathcal{P} \subseteq \mathcal{V}$ to visit a set of n customers $\mathcal{C} \subseteq \mathcal{V}$, find a set of m truck group routes $\{\pi_p\}_{p\in\mathcal{P}}$, one for each truck group, to visit every customer $c \in \mathcal{C}$ exactly once that minimizes the total cost $\sum_{p\in\mathcal{P}} cost(\pi_p)$, while satisfying constraints 1∼4.*

We provide a MILP formulation in Appendix C.

3 Methodology

Algorithm 1: Framework to solve MA-FSTSP

Require: Road network $\mathcal{G}$, distance d, customers $\mathcal{C}$, depots $\mathcal{P}$, truck speed s_{tr}, drone speed s_{dr}, drone endurance r

1: Initialize solution SOLN as empty set
2: **for** $(u, v) \in \mathcal{V}(\mathcal{G}) \times \mathcal{V}(\mathcal{G})$ **do**
3: $\quad d^{\text{tr}}(u, v) \leftarrow$ ShortestPathLength$(u, v; \mathcal{G})$
4: **end for**
5: Assign customers $\{\mathcal{C}_i\}_{i=1}^{|\mathcal{P}|} \leftarrow$ Partition$(\mathcal{C}, \mathcal{P}; d^{\text{tr}}, d)$
6: **for** $i \in \{1, 2, \cdots, m\}$ **do**
7: $\quad \mathcal{S} \leftarrow \bigcup_{c\in C_i}\{\{v \in \mathcal{V}(\mathcal{G}) : d(v, c) < r/2\}\} \cup \{\{p_i\}\}$
8: $\quad$ ORDER $\leftarrow$ SetTSP$(\mathcal{S}; d^{\text{tr}}, d; s_{\text{tr}}, s_{\text{dr}})$
9: $\quad$ SOLN$[i] \leftarrow$ GetRoute$(\mathcal{C}_i, p_i;$ ORDER$; d^{\text{tr}}, d; s_{\text{tr}}, s_{\text{dr}})$
10: **end for**
11: **return** SOLN

We propose a three-phase algorithm to solve MA-FSTSP, as shown in Alg. 1. In phase 1, the problem is decomposed into m subproblems (recall that m is the number of depots) of a single truck carrying multiple drones to visit customers on road networks by allocating customers to truck groups via set-based partition algorithms. Next, in phase 2, a TSP heuristic is applied to determine the customer visiting order for each truck group. Finally, in phase 3, a truck-drone route is computed, given the restriction of customers' visiting orders. The pseudocode for the proposed algorithm is shown in Alg. 1. Lines 1∼4 are the preparation for the data. Line 5 corresponds to phase 1. Line 7 computes the set for each customer assigned to depot p_i, and lines 8∼9 correspond to phases 2 and 3, respectively. Methods of phase 1∼3 are introduced in Sec. 3.1, Sec. 3.2, and Sec. 3.3, respectively. An illustration, an acceleration method, and the computational complexity analysis are introduced in the supplementary material. An example is shown in Fig. 1.

3.1 Phase 1: Set-based partition algorithms

The first phase of our algorithm introduces a novel set-based heuristic for assigning customers to truck groups, addressing key limitations in existing truck-drone delivery systems.

Traditional customer assignment methods in multi-agent systems include the Nearest-Neighbor (NN) algorithm (Ho et al. 2008; Salhi, Imran, and Wassan 2014; Geetha, Vanathi, and Poonthalir 2012) and the Minimum Spanning Tree (MST) partition algorithm (Yang and Fan 2024). However, these methods often fall short in practical scenarios where geometric proximity fails to accurately represent the traveling time or cost. Consider two geographically adjacent locations separated by a highway or river - while physically close, they may require significant travel time to reach one another. Similar challenges arise with one-way roads and traffic bottlenecks. In such cases, neither geometric nor road map distance provides an accurate estimate of the time required for truck groups to serve customers.

To address these limitations, we propose a set-based distance metric that more precisely approximates the pointwise distance for truck groups. We extend the NN and MST partitions by substituting our set-based distance metric for the geometric/Euclidean distance measurement.

For a customer $c \in \mathcal{C}$, we define its set of neighbor vertices:

$$\mathcal{S}_c(\theta) \coloneqq \{n \in \mathcal{V}(\mathcal{G}) : d(n, c) \le \theta\} \tag{1}$$

where θ is a distance parameter, and $d(n, c)$ represents the distance between vertex n and customer c. It is worth noting that if $\theta \le r$, each vertex in $\mathcal{S}_c$ serves as a valid takeoff or landing point for a drone visiting customer c. Using this set, we define the distance from set $\mathcal{S}_c(\theta)$ to set $\mathcal{S}_{c'}(\theta)$ as

$$\begin{aligned} &d^{\text{set}}(\mathcal{S}_c(\theta), \mathcal{S}_{c'}(\theta)) \\ \coloneqq &\min_{v\in\mathcal{S}_c(\theta), v'\in\mathcal{S}_{c'}(\theta)} \left\{ \frac{d(c, v)}{s_{\text{dr}}} + \frac{d^{\text{tr}}(v, v')}{s_{\text{tr}}} + \frac{d(v', c')}{s_{\text{dr}}} \right\} \cdot s_{\text{tr}}, \end{aligned} \tag{2}$$

where d^{tr} represents the shortest path length for the truck, and s_{dr} and s_{tr} are the speeds of the drone and truck, respectively. This distance metric is normalized based on the time required for a truck group to visit customer c' after visiting customer c. Since the vertex p can also be viewed as a set $\{p\} = S_p(0^+)$ containing p only, we extend the domain of $\mathcal{S}_c$ from $c \in \mathcal{C}$ to $c \in \mathcal{C} \cup \mathcal{P}$. Without ambiguity, we omit θ and abbreviate the distance as $d^{\text{set}}(c, c')$ for $c, c' \in \mathcal{C} \cup \mathcal{P}$.

Algorithm 2: Pseudocode for Set-Based Paritition

Require: Road network $\mathcal{G}$, distance d, customers $\mathcal{C}$, depots $\mathcal{P}$, truck speed s_{tr}, drone speed s_{dr}, radius θ

1: **for** $c \in \mathcal{C}$ **do**
2: $\quad \mathcal{S}_c(\theta) \leftarrow \{v \in \mathcal{V}(\mathcal{G}) : d(v, c) \le \theta\}$
3: **end for**
4: **for** $p \in \mathcal{P}$ **do**
5: $\quad \mathcal{S}_p \leftarrow \{p\}$
6: **end for**
7: Construct fully connected graph $\mathcal{G}'(\mathcal{C} \cup \mathcal{P})$ by edge weight $w(c, c') = d^{\text{set}}(c, c')$ {Use Eq. 2}
8: $\{\mathcal{C}_p\}_{p\in\mathcal{P}} \leftarrow$ NNPartition$(\mathcal{P}, \mathcal{C})$ or MSTPartition$(\mathcal{P}, \mathcal{C})$
9: **return** assignment $\{\mathcal{C}_p\}_{p\in\mathcal{P}}$

Alg. 2 presents the implementation of our set-based partition approach. The algorithm first formulates the set (lines

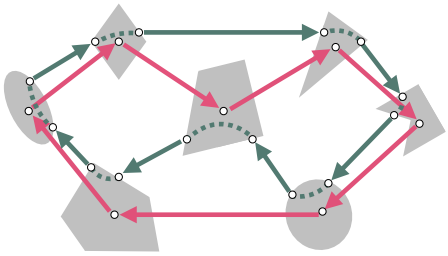


Figure 3: An example of Set TSP versus TSP. Each gray set represents a set of vertices around the central vertex (customer), which can be connected arbitrarily within. Sets are fully connected via vertices at the boundary, which is omitted. The red arrows represent the optimal TSP tour to visit the customers, and the green arrows represent the optimal Set TSP tour to visit every set. Since Set TSP does not require visiting any exact vertex, it can pass through the set via **shortcuts** represented in dashed green. So, the visiting order of sets in the TSP tour and the Set TSP tour can differ.

1-6) to construct a fully connected graph (line 7). It then applies either the nearest-neighbor or MST partition algorithm to determine the final customer assignments (line 8). Following this initial assignment, each truck group optimizes its tours in phases 2-3.

3.2 Phase 2: Set TSP Heuristic

Following the same insight in Sec. 3.1, we extend the TSP to a set-based version that accounts for drone operations. This extension is formalized as follows:

Definition 2 (Set TSP, STSP). *Given n location **sets** $V_1, V_2, \cdots, V_n$ and the traveling costs between each pair of locations, find the route with the lowest total cost to visit each location set exactly once and return to the starting location set. A visit to a location set is defined as consecutively visiting one or more locations within the set.*

An example of Set TSP versus TSP is shown in Fig. 3.

For a truck group $\mathcal{T}_i$, its starting depot p_i, and assigned customers $\mathcal{C}_i$, we form the following Set TSP:

1. Collect $|\mathcal{C}_i|+1$ location sets $S_c(\frac{r}{2}), c \in \mathcal{C}_i$ and $\{p_i\}$
2. The traveling cost from $u \in S_c(\theta), c \in C_i$ to v is

$$w(u,v) = \begin{cases} d^{\text{tr}}(u,v)/s_{\text{tr}}, & v \notin S_c(\theta); \\ \tilde{w}(c;u,v), & v \in S_c(\theta). \end{cases} \tag{3}$$

where $d^{\text{tr}}(u,v)$ is the shortest path length from u to v,

$$\tilde{w}(c;u,v) = \min\Big\{ \max\Big\{ \frac{d(u,c)+d(c,v)}{s_{\text{dr}}}, \frac{d(u,v)}{s_{\text{tr}}} \Big\}, \frac{d^{\text{tr}}(u,c)+d^{\text{tr}}(c,v)}{s_{\text{tr}}} \Big\}, \tag{4}$$

which is the time required for the truck group to visit customer c, with the drone departing from vertex u and landing at vertex v.

The solution of the Set TSP provides a feasible tour for the truck group by departing the drone at the location entering the $S_c(\frac{r}{2})$ and collecting it at the location leaving the $S_c(\frac{r}{2})$. The drone flys at most $\frac{r}{2}+\frac{r}{2}=r$, which will not exceed the battery limit. The visiting sequence of customers in the solution is used in Phase 3.

Since the Set TSP is NP-hard as it generalizes the classic TSP, we develop an extended version of the Gavish-Graves (GG) formula (Gavish and Graves 1978).

Let $\beta_{c,c'} \in \{0,1\}$ be a binary variable indicating whether the next customer or depot is c' after visiting customer or depot c, and $y_{c,c'} \in \mathbb{R}_+$ be the corresponding network flow. We ensure the TSP property by:

$$\beta_{c,c} = y_{c,c} = 0, \forall c \in \mathcal{C}_i \cup \{p_i\}, \tag{5a}$$

$$\sum_{c' \in \mathcal{C}_i \cup \{p_i\}} \beta_{c,c'} = \sum_{c' \in \mathcal{C}_i \cup \{p_i\}} \beta_{c',c} = 1, \forall c \in \mathcal{C}_i \cup \{p_i\}, \tag{5b}$$

$$y_{c,c'} \le |\mathcal{C}_i| \cdot \beta_{c,c'}, \forall c, c' \in \mathcal{C}_i \cup \{p_i\}, \tag{5c}$$

$$\sum_{c \in \mathcal{C}_i \cup \{p_i\}} y_{p_i,c} = |\mathcal{C}_i|, \tag{5d}$$

$$\sum_{c \in \mathcal{C}_i \cup \{p_i\}} y_{c,p_i} = 0, \tag{5e}$$

$$\sum_{c' \in \mathcal{C}_i \cup \{p_i\}} y_{c',c} - \sum_{c' \in \mathcal{C}_i \cup \{p_i\}} y_{c,c'} = 1, \forall c \in \mathcal{C}_i \cup \{p_i\}. \tag{5f}$$

The Eq. 5a forbids self-loops, Eq. 5b forces the solution to be a collection of cycles, and Eq. 5c– Eq. 5f ask the flow to reduce 1 unit each step along the cycle from $|\mathcal{C}_i|$ to 0, which only allows the existence of one cycle.

Let $\gamma_{u,v}$ be a binary variable indicating whether a drone tour starts from u to v for $u, v \in \mathcal{S}_c$ and $\delta_{v,u}$ be a binary variable indicating whether a truck collects a drone at v and carries it to u to launch it. The constraints are:

$$\sum_{u,v \in \mathcal{S}_c} \gamma_{u,v} = 1, \forall c \in \mathcal{C}_i, \tag{6a}$$

$$\sum_{v \in \mathcal{S}_c} \sum_{u \in \mathcal{S}_{c'}} \delta_{v,u} = \beta_{c,c'}, \forall c, c' \in \mathcal{C}_i \cup \{p_i\}, \tag{6b}$$

$$\sum_{u \in \mathcal{S}_c} \gamma_{u,v} = \sum_{c' \in \mathcal{C}_i \cup \{p_i\}} \sum_{w \in \mathcal{S}_{c'}} \delta_{v,w}, \forall c \in \mathcal{C}_i \cup \{p_i\}, v \in \mathcal{S}_c \tag{6c}$$

$$\sum_{v \in \mathcal{S}_c} \gamma_{u,v} = \sum_{c' \in \mathcal{C}_i \cup \{p_i\}} \sum_{w \in \mathcal{S}_{c'}} \delta_{w,u}, \forall c \in \mathcal{C}_i \cup \{p_i\}, u \in \mathcal{S}_c. \tag{6d}$$

Eq. 6a ensures each customer receives exactly one drone visit. Eq. 6b aligns the vertex- and set-level routes. If c is visited followed by the visit of c', i.e., $\beta_{c,c'} = 1$, then the truck route should pass from a collection location $v \in S_c$ to a new launching location $u \in S_{c'}$. Otherwise, such a subroute does not exist in the truck route. Eq. 6c means that if the drone lands at $v \in S_c$ after visiting c, it is collected at v by the truck. It synchronizes the truck and the drone at the landing vertex. Similarly, Eq. 6d synchronizes the truck and the drone at the launching vertex.

The objective function minimizes the sum of costs of the edges (defined in Eq. 3) traversed by the route

$$\sum_{c \in \mathcal{C}_i \cup \{p_i\}} \sum_{u,v \in \mathcal{S}_c} w(u,v) \cdot \gamma_{u,v} + \sum_{c,c' \in \mathcal{C}_i \cup \{p_i\}} \sum_{u \in \mathcal{S}_c} \sum_{v \in \mathcal{S}_{c'}} w(u,v) \cdot \delta_{u,v}. \tag{7}$$

γ and δ together form the adjacent matrix of the truck group tour on the fully connected graph $\mathcal{G}(\{p_i\} \cup_{c \in \mathcal{C}_i} S_c(\theta))$, from which we can extract the order of customers to visit.

3.3 Phase 3: Dynamic Programming for Truck-Drone Optimization

Even with a predetermined customer visiting order, finding the optimal route for a truck group remains an NP-hard problem (Tang, Hoeve, and Shaw 2019). To develop a tractable solution, we introduce an approximation that considers simultaneous drone dispatches. The approach is effective when r is small, as multi-drone operations only benefit when a cluster of customers is within $O(r)$ of each other.

Under the assumption of a simultaneous dispatch, we develop a dynamic programming solution that computes the optimal route in polynomial time. For truck group $\mathcal{T}_i$ and the given visiting sequence $\mathcal{O}_i = \{o_i^1, o_i^2, \cdots, o_i^{|\mathcal{C}_i|}\}$ of assigned customers $\mathcal{C}_i$, we define two key functions:

1. $\textsc{Time}(u, v; s, t)$ represents the minimum time required for the truck group to serve customers $o_i^s, o_i^{s+1}, \cdots, o_i^{s+t-1}$ using t drones launched simultaneously from u, with the final drone collection at v.
2. $\textsc{Value}(s, v)$: The optimal cost from depot p_i to vertex v after serving the first s customers.

For the base case of a single drone ($t = 1$), we define

$$\textsc{Time}(u, v; s, 1) = \begin{cases} \dfrac{d^{\text{tr}}(u, o_i^s) + d^{\text{tr}}(o_i^s, v)}{s_{\text{tr}}}, & d(u, o_i^s) + d(o_i^s, v) > r; \\ \min\Big\{ \max\Big\{ \dfrac{d(u, o_i^s) + d(o_i^s, v)}{s_{\text{dr}}}, \dfrac{d^{\text{tr}}(u, v)}{s_{\text{tr}}} \Big\}, & \\ \quad \dfrac{d^{\text{tr}}(u, o_i^s) + d^{\text{tr}}(o_i^s, v)}{s_{\text{tr}}} \Big\}, & \text{otherwise.} \end{cases} \tag{8}$$

The first case, $d(u, o_i^s) + d(o_i^s, v) > r$, handles the situation where drone delivery is infeasible due to range limitations, defaulting to truck delivery. The second case computes the minimum time considering both drone flight and truck travel times when drone delivery is possible.

For multiple drones ($t \geq 2$), to simplify the formulation, we denote $S' = S_{o_i^{s+t-2}}(r)$ to be the set of nodes within the range of distance r to the order o_i^{s+t-2}, and $o_i^{-1} = o_i^{s+t-1}$ to be the last order. We compute the update

$$\begin{aligned} &\textsc{Time}(u, v; s, t) = \min\{ \\ &\min_{w \in S'} \textsc{Time}(u, w; s, t-1) + (d^{\text{tr}}(w, o_i^{-1}) + d^{\text{tr}}(o_i^{-1}, v))/s_{\text{tr}}, \\ &\max\{ \min_{w \in S'} \textsc{Time}(u, w; s, t-1) + d^{\text{tr}}(w, v)/s_{\text{tr}}, \\ &\qquad (d(u, o_i^{-1}) + d(o_i^{-1}, v))/s_{\text{dr}} \}\}, \end{aligned} \tag{9}$$

when $\textsc{Time}(u, w; s, t-1)$ is finite for all $w \in S_{o_i^{s+t-2}}(r)$, and the last drone delivery is feasible; otherwise, we set $\textsc{Time}(u, v; s, t)$ to be $+\infty$.

The VALUE function initialization is defined as:

$$\textsc{Value}(0, v) = d^{\text{tr}}(p_i, v)/s^{\text{tr}}, \forall v \in \cup_{c \in \mathcal{O}_i} \mathcal{S}_c(r), \tag{10}$$

which is the shortest time from depot p_i to vertex v. We restrict vertex initialization to $\cup_{c \in \mathcal{O}_i} \mathcal{S}_c(r)$, as drones must operate within their battery capacity limits.

For a truck group with k drones, the transition function takes the k preceding states into account, which is

$$\begin{aligned} \textsc{Value}(s, v) = &\min_{0 \leq t \leq k,\ u \in \mathcal{S}_{o_i^{s-t}}(r), w \in \mathcal{S}_{o_i^s}(r)} d^{\text{tr}}(w, v)/s^{\text{tr}} \\ &+ \textsc{Time}(u, w; s-t+1, t) + \textsc{Value}(s-t, u). \end{aligned} \tag{11}$$

Here, we also reduce the search space for vertices u and w to the relevant neighborhoods defined by the drone's range constraints to eliminate unnecessary computation. The search space can be further reduced to neighborhood boundaries to accelerate the computation. The details of the acceleration are in the supplementary materials. We also omit calculating all intermediate truck-moving actions since they yield consistent values along optimal routes.

The optimal cost for truck group $\mathcal{T}_i$ is given by $\textsc{Value}(|\mathcal{C}_i|, p_i)$, and the corresponding route can be reconstructed by backward tracing of the maximum conditions.

4 Experiments and Analysis

In this section, we report experiments to validate our method. In Sec. 4.1, we show the effectiveness and scalability by comparing our method with four baselines: Column Generation (CG) (Gao et al. 2023), greedy, Ant Colony Optimization (ACO), and a Variable Neighborhood Search (VNS). In Sec. 4.2, we explore the scalability of our approach by varying the problem size in different ways. We validate the set extension approach in Sec. 4.3. In Sec. 4.4, we do the sensitivity analysis to explore the influence of drone speed and the distance limit. The sensitivity analysis of the number of drones per truck is in Appendix D.

All experiments are performed on a server running on Ubuntu 20.04.6 LTS with an AMD Ryzen Threadripper 3990X 64-Core Processor. The MILP is solved by Gurobi 10.0.2 (Gurobi Optimization, LLC 2023) with a time limit of 7200 seconds. All algorithms are implemented in Python.

4.1 Comparison against Baselines

We evaluate our proposed approach against multiple baseline methods across three real-world road networks of varying scales. All experiments measure both solution quality (delivery cost) and computational efficiency (runtime).

Datasets. We utilize three road network datasets of increasing complexity. On each road network, we randomly generate three datasets, with 100 instances each, of different numbers of customers:

1. Partial Manhattan: A strongly connected component with 20 vertices. Generate three datasets with 2 depots and 5/10/15 customers.
2. Complete Manhattan: The full Manhattan road network with 1,024 vertices (Blahoudek et al. 2020). Generate three datasets with 5 depots and 50/100/150 customers.
3. Boston: OpenStreetMap-derived network with 11,000 vertices (OpenStreetMap contributors 2017). Generate three datasets with 10 depots and 50/100/150 customers.

The Boston network presents additional challenges due to its higher density—the same radius covers more nodes than in Manhattan, increasing problem complexity even with identical customer numbers.

	$\|\mathcal{C}\| = 5$		$\|\mathcal{C}\| = 10$		$\|\mathcal{C}\| = 15$	
Alg.	Cost	Time	Cost	Time	Cost	Time
CG	3.69	727.81	4.61	1259.44	6.25	2026.17
Greedy	1.26	**0.00**	2.08	**0.01**	2.74	**0.02**
ACO	1.15	1.05	1.54	2.36	1.81	3.75
VNS	0.87	0.49	1.53	0.99	1.98	1.47
Ours	**0.74**	0.07	**1.17**	0.26	**1.47**	0.47

(a) The partial Manhattan road network. Time in seconds.

	$\|\mathcal{C}\| = 50$		$\|\mathcal{C}\| = 100$		$\|\mathcal{C}\| = 150$	
Alg.	Cost	Time	Cost	Time	Cost	Time
CG	-	-	-	-	-	-
Greedy	32.08	0.64	47.20	2.30	57.08	5.17
ACO	41.12	22.70	67.02	63.15	87.00	111.03
VNS	34.18	6.09	49.24	13.15	59.70	22.33
Ours	**18.74**	**0.58**	**24.04**	**1.51**	**27.89**	**3.23**

(b) The full Manhattan road network. Time in minutes.

	$\|\mathcal{C}\| = 50$		$\|\mathcal{C}\| = 100$		$\|\mathcal{C}\| = 150$	
Alg.	Cost	Time	Cost	Time	Cost	Time
CG	-	-	-	-	-	-
Greedy	90.32	5.33	134.95	18.38	162.12	38.79
ACO	-	-	-	-	-	-
VNS	106.13	11.44	146.53	42.84	176.93	83.99
Ours	**72.57**	**1.63**	**90.54**	**8.40**	**99.64**	**21.42**

(c) The Boston road network. Time in minutes.

Table 1: The performance comparison against baselines. The best performances are marked in bold.

Baselines. We compare our method against four baselines:

1. **Column Generation (CG)** (Gao et al. 2023) is a heuristic method that decomposes the problem into master (partitioning) and pricing (routing) problems, using VNS for approximation.
2. **Greedy** is a custom implementation that iteratively selects the customer-truck pair to minimize the one-step completion time.
3. **Ant Colony Optimization (ACO)** (Dorigo and Gambardella 2002) is a meta-heuristic method that uses pheromone-based selection to determine the next customer and updates pheromones based on the reciprocals of completion times.
4. **Variable Neighborhood Search (VNS)** is a custom implementation that combines nearest-neighbor assignment with Hill-Climbing (Chinnasamy et al. 2022), using three neighborhood operations: (1) modifying drone takeoff/landing locations; (2) reassigning customers between truck and drone service; (3) swapping consecutive customer visit orders.

We excluded genetic algorithm and particle swarm optimization (Lin et al. 2022) due to their inability to solve even the smallest test cases within reasonable time limits.

Parameters. We set the number of drones on each truck to 2 (Partial Manhattan), 3 (Complete Manhattan), and 4 (Boston). The truck speed is set to 30 km/h and the drone speed to 48 km/h (Gao et al. 2023). The drone delivery distance limit r is set to 1.5 km.

	$\|\mathcal{C}\| = 50$		$\|\mathcal{C}\| = 100$		$\|\mathcal{C}\| = 150$	
Method	Cost	Time	Cost	Time	Cost	Time
NN + TSP	21.89	26.18	28.89	60.64	33.78	99.01
NN + STSP	21.17	28.49	27.08	67.45	30.85	115.01
MST + TSP	19.51	27.05	25.82	62.59	30.94	100.80
MST + STSP	19.00	29.95	24.21	72.18	28.05	126.87
SNN + TSP	21.38	26.52	28.31	61.17	33.01	99.31
SNN + STSP	20.55	31.14	26.15	77.02	29.78	141.29
SMST + TSP	19.43	27.09	25.71	62.35	30.87	100.83
SMST + STSP	**18.74**	34.84	**24.04**	90.82	**27.89**	193.71

Table 2: Results averaged over 100 cases sampled uniformly from Manhattan per scenario. Minimum cost marked bold. Time reported in seconds.

Results. As shown in Tab. 1, our method demonstrates superior performance across all test scenarios. Our method provides the minimum cost solutions in all test scenarios. The method shows superior performance on medium- and large-scale problems, achieving more than 30% cost reduction over the next-best method. The computational efficiency of our method is also impressive. While the greedy algorithm performs faster in small instances, our method maintains the fastest execution time in medium and larger instances, where other methods struggle or fail to produce solutions. These comprehensive results validate the effectiveness of our approach for coordinated truck-drone delivery systems, combining high-quality solutions with practical computational efficiency for real-world applications.

4.2 Scalability Analysis

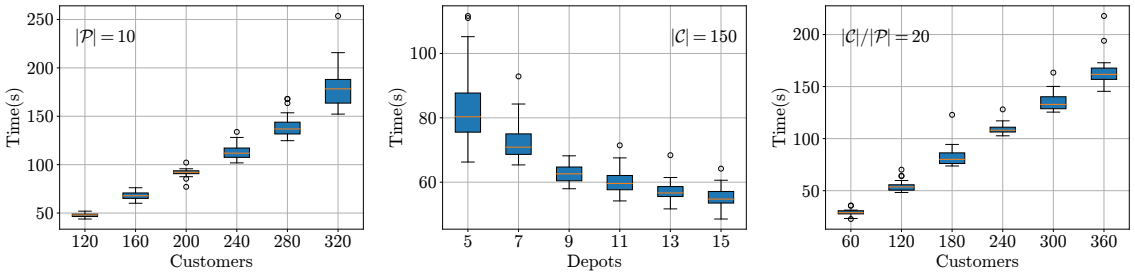
We evaluate the scalability of our proposed methods by analyzing how computational time varies with problem size. Using the set-NN algorithm in Phase 1 and the set-TSP algorithm in Phase 2, we conduct three experiments examining the relationship between computational time and both the number of customers $|\mathcal{C}|$ and the number of depots $|\mathcal{P}|$.

The results in Fig. 4 demonstrate that our algorithm effectively handles problems with up to 300 customers. With fixed $|\mathcal{P}|$ (Fig. 4a), computational time increases exponentially with $|\mathcal{C}|$. When $|\mathcal{C}|$ is fixed (Fig. 4b), the computational time shows almost a $1/|\mathcal{P}|$ relationship. Under a constant $|\mathcal{C}|/|\mathcal{P}|$ ratio (Fig. 4c), we observe near-linear growth in computational time as both parameters increase.

These patterns indicate that the Set-TSP problem in Phase 2 represents the primary computational bottleneck, suggesting a clear direction for future algorithmic optimizations.

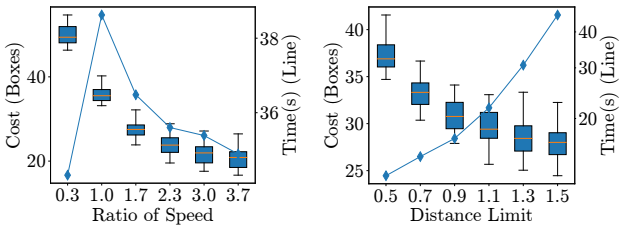
4.3 Effectiveness of Set-based Methods

We analyze the effectiveness of our set extension approach by comparing different algorithmic combinations across



(a) Fix the number of depots $|\mathcal{P}|$ (b) Fix the number of customers $|\mathcal{C}|$ (c) Fix the customer-depot ratio $|\mathcal{C}|/|\mathcal{P}|$

Figure 4: Box plots of scalability experiments.



(a) Relative speed $s_{\text{dr}}/s_{\text{tr}}$ (b) Distance limit r

Figure 5: The influence of drone speed and distance limit. Time in Fig. (b) is plotted on a logarithmic scale.

both phases. The first phase evaluates nearest neighbor (NN), minimum spanning tree (MST), and their set-based extensions (SNN, SMST). The second phase compares the standard TSP with the Set TSP (STSP). We test all combinations on three datasets, each containing 100 instances from the Manhattan road network with 5 depots and different numbers of customers (50, 100, or 150).

As shown in Tab. 2, SMST + STSP consistently achieves the lowest cost across all problem sizes, demonstrating the effectiveness of the set-based method, though at the expense of increased computational time.

We also find some key patterns in the results. First, STSP consistently outperforms standard TSP in solution quality across all first-phase methods, with the improvement becoming more significant as the number of customers increases. This advantage comes at the cost of increased computational time, particularly for large instances.

Second, MST-based methods (both standard and set versions) produce better solutions than NN-based approaches, with higher computational cost. This increased computation time stems from MST's more unbalanced customer allocation, underscoring the bottleneck in the second phase.

Third, set-based methods in Phase 1 show interesting interaction effects with Phase 2 choices. When paired with standard TSP, set-based first-phase methods offer only a limited advantage over their traditional counterparts. However, when combined with STSP, they achieve remarkably better solutions, suggesting the importance of using consistent distance metrics across both phases.

4.4 Sensitivity Analysis

We analyze how drone parameters influence system performance, including speed, flying distance limit, and fleet size. Results for fleet size are in Appendix D.

Speed. The experiment is conducted on the Manhattan dataset of 5 depots and 100 customers. Fig. 5a shows the effect of varying drone speed s_{dr} from 10 km/h to 110 km/h. Delivery cost decreases approximately proportionally to $1/s_{\text{dr}}$, demonstrating that drones provide benefits even at lower speeds because they can bypass road networks. We also observe a peak in computational time when drone and truck speeds are similar, as more comparable vertex pairs increase the MILP solution space.

Distance limit. As shown in Fig. 5b, increasing the drone delivery range r from 0.5 km to 1.5 km results in sub-linear cost reduction, constrained by the drone fleet size and drone speed. However, computational time grows exponentially with r due to the linear increase in the boundary vertices.

5 Conclusion

In this paper, we introduce and solve the Multi-Agent Flying Sidekick Traveling Salesman Problem (MA-FSTSP), a novel extension of the FSTSP that addresses real-world challenges in coordinated truck-drone delivery systems. Our key contributions include problem formulation and an efficient three-phase solution that leverages set-based heuristics to address the complexities of practical routing scenarios.

Our comprehensive experimental results demonstrate the method's effectiveness across multiple dimensions. It consistently outperforms existing baselines, reducing solution costs by more than 30% while keeping computational efficiency. Scalability tests show the method handles problems with hundreds of customers while maintaining solution quality. Sensitivity analyses reveal that our method effectively leverages drone advantages even at a low speed.

We identify two promising directions for future research. First, the computational bottleneck in the Set-TSP could be addressed by adapting heuristic algorithms like the Lin-Kernighan Heuristic (LKH) (Lin and Kernighan 1973), potentially improving scalability without compromising solution quality. Second, the discrete vertex-based setting could be extended to allow continuous drone operations along road edges to produce more efficient delivery strategies.

Acknowledgments

This work was partly supported by the National Science Foundation (NSF) CAREER Award #CCF-2238030 and the MITEI Future Energy Systems Center. Any opinions, findings, conclusions, or recommendations expressed in this publication are those of the authors and don't necessarily reflect the views of the sponsors.

References

Blahoudek, F.; Brázdil, T.; Novotnỳ, P.; Ornik, M.; Thangeda, P.; and Topcu, U. 2020. Qualitative controller synthesis for consumption Markov decision processes. In *International Conference on Computer Aided Verification*, 421–447. Springer.

Bruni, M. E.; Khodaparasti, S.; and Moshref-Javadi, M. 2022. A logic-based Benders decomposition method for the multi-trip traveling repairman problem with drones. *Computers & Operations Research*, 145: 105845.

Carlsson, J. G.; and Song, S. 2018. Coordinated logistics with a truck and a drone. *Management Science*, 64(9): 4052–4069.

Cavani, S.; Iori, M.; and Roberti, R. 2021. Exact methods for the traveling salesman problem with multiple drones. *Transportation Research Part C: Emerging Technologies*, 130: 103280.

Chen, C.; Demir, E.; and Huang, Y. 2021. An adaptive large neighborhood search heuristic for the vehicle routing problem with time windows and delivery robots. *European journal of operational research*, 294(3): 1164–1180.

Chiang, W.-C.; Li, Y.; Shang, J.; and Urban, T. L. 2019. Impact of drone delivery on sustainability and cost: Realizing the UAV potential through vehicle routing optimization. *Applied energy*, 242: 1164–1175.

Chinnasamy, S.; Ramachandran, M.; Amudha, M.; and Ramu, K. 2022. A review on hill climbing optimization methodology. *Recent Trends in Management and Commerce*, 3(1).

Choudhury, S.; Solovey, K.; Kochenderfer, M.; and Pavone, M. 2021a. Coordinated multi-agent pathfinding for drones and trucks over road networks. *arXiv preprint arXiv:2110.08802*.

Choudhury, S.; Solovey, K.; Kochenderfer, M. J.; and Pavone, M. 2021b. Efficient large-scale multi-drone delivery using transit networks. *Journal of Artificial Intelligence Research*, 70: 757–788.

Dorigo, M.; and Gambardella, L. M. 2002. Ant colony system: a cooperative learning approach to the traveling salesman problem. *IEEE Transactions on evolutionary computation*, 1(1): 53–66.

Gao, J.; Zhen, L.; Laporte, G.; and He, X. 2023. Scheduling trucks and drones for cooperative deliveries. *Transportation Research Part E: Logistics and Transportation Review*, 178: 103267.

Gao, J.; Zhen, L.; and Wang, S. 2023. Multi-trucks-and-drones cooperative pickup and delivery problem. *Transportation Research Part C: Emerging Technologies*, 157: 104407.

Gavish, B.; and Graves, S. C. 1978. The travelling salesman problem and related problems.

Geetha, S.; Vanathi, P.; and Poonthalir, G. 2012. Metaheuristic approach for the multi-depot vehicle routing problem. *Applied Artificial Intelligence*, 26(9): 878–901.

Greshler, N.; Gordon, O.; Salzman, O.; and Shimkin, N. 2021. Cooperative multi-agent path finding: Beyond path planning and collision avoidance. In *2021 International Symposium on Multi-Robot and Multi-Agent Systems (MRS)*, 20–28. IEEE.

Gurobi Optimization, LLC. 2023. Gurobi Optimizer Reference Manual.

Ho, W.; Ho, G. T.; Ji, P.; and Lau, H. C. 2008. A hybrid genetic algorithm for the multi-depot vehicle routing problem. *Engineering applications of artificial intelligence*, 21(4): 548–557.

Karak, A.; and Abdelghany, K. 2019. The hybrid vehicle-drone routing problem for pick-up and delivery services. *Transportation Research Part C: Emerging Technologies*, 102: 427–449.

Kitjacharoenchai, P.; Ventresca, M.; Moshref-Javadi, M.; Lee, S.; Tanchoco, J. M.; and Brunese, P. A. 2019. Multiple traveling salesman problem with drones: Mathematical model and heuristic approach. *Computers & Industrial Engineering*, 129: 14–30.

Li, H.; Chen, J.; Wang, F.; and Zhao, Y. 2022. Truck and drone routing problem with synchronization on arcs. *Naval Research Logistics (NRL)*, 69(6): 884–901.

Lin, M.; Chen, Y.; Han, R.; Chen, Y.; et al. 2022. Discrete optimization on truck-drone collaborative transportation system for delivering medical resources. *Discrete Dynamics in Nature and Society*, 2022.

Lin, S.; and Kernighan, B. W. 1973. An effective heuristic algorithm for the traveling-salesman problem. *Operations research*, 21(2): 498–516.

Lu, Y.; Yang, C.; and Yang, J. 2022. A multi-objective humanitarian pickup and delivery vehicle routing problem with drones. *Annals of Operations Research*, 319(1): 291–353.

Luo, Z.; Gu, R.; Poon, M.; Liu, Z.; and Lim, A. 2022. A last-mile drone-assisted one-to-one pickup and delivery problem with multi-visit drone trips. *Computers & Operations Research*, 148: 106015.

Mbiadou Saleu, R. G.; Deroussi, L.; Feillet, D.; Grangeon, N.; and Quilliot, A. 2018. An iterative two-step heuristic for the parallel drone scheduling traveling salesman problem. *Networks*, 72(4): 459–474.

Murray, C. C.; and Chu, A. G. 2015. The flying sidekick traveling salesman problem: Optimization of drone-assisted parcel delivery. *Transportation Research Part C: Emerging Technologies*, 54: 86–109.

Murray, C. C.; and Raj, R. 2020. The multiple flying sidekicks traveling salesman problem: Parcel delivery with multiple drones. *Transportation Research Part C: Emerging Technologies*, 110: 368–398.

OpenStreetMap contributors. 2017. Planet dump retrieved from https://planet.osm.org . https://www.openstreetmap.org.

Poikonen, S.; Wang, X.; and Golden, B. 2017. The vehicle routing problem with drones: Extended models and connections. *Networks*, 70(1): 34–43.

Sacramento, D.; Pisinger, D.; and Ropke, S. 2019. An adaptive large neighborhood search metaheuristic for the vehicle routing problem with drones. *Transportation Research Part C: Emerging Technologies*, 102: 289–315.

Salama, M. R.; and Srinivas, S. 2022. Collaborative truck multi-drone routing and scheduling problem: Package delivery with flexible launch and recovery sites. *Transportation Research Part E: Logistics and Transportation Review*, 164: 102788.

Salhi, S.; Imran, A.; and Wassan, N. A. 2014. The multi-depot vehicle routing problem with heterogeneous vehicle fleet: Formulation and a variable neighborhood search implementation. *Computers & Operations Research*, 52: 315–325.

Tamke, F.; and Buscher, U. 2021. A branch-and-cut algorithm for the vehicle routing problem with drones. *Transportation Research Part B: Methodological*, 144: 174–203.

Tang, Z.; Hoeve, W.-J. v.; and Shaw, P. 2019. A study on the traveling salesman problem with a drone. In *Integration of Constraint Programming, Artificial Intelligence, and Operations Research: 16th International Conference, CPAIOR 2019, Thessaloniki, Greece, June 4–7, 2019, Proceedings 16*, 557–564. Springer.

Yang, R.; and Fan, C. 2024. A Hierarchical Framework for Solving the Constrained Multiple Depot Traveling Salesman Problem. *IEEE Robotics and Automation Letters*.