Departamento de
Ingeniería Industrial
Universidad de Concepción

UNA MATHEURÍSTICA PARA EL PROBLEMA DEL VENDEDOR VIAJERO CON MÚLTIPLES DRONES

Por: Benjamín Brandt Mieres

Tesis presentada a la Facultad de Ingeniería de la Universidad de Concepción para optar al grado académico de Magíster en Ingeniería Industrial

Septiembre 2024
Concepción, Chile

Profesor Guía: Dr. Carlos Contreras Bolton

© 2024, Benjamín Brandt Mieres
Se autoriza la reproducción total o parcial, con fines académicos, por cualquier medio o procedimiento, incluyendo la cita bibliográfica del documento.

ACKNOWLEDGMENTS

Quiero agradecer a mis padres, hermanos y familiares por su constante apoyo a lo largo de mi vida. Sus consejos y aliento han sido fundamentales en mi desarrollo. También, agradecer a mis amigos por compartir risas y alegrías en cada paso del camino, haciendo que incluso los momentos más desafiantes se conviertan en buenos recuerdos. Finalmente, agradecer a mis profesores por su dedicación en la enseñanza y brindarme herramientas valiosas para crecer.

Esta tesis fue parcialmente apoyada por la infraestructura de supercómputo del NLHPC (ECM-02) y por ANID mediante el FONDECYT INICIACIÓN 11241132.

Resumen

Diferentes investigaciones demuestran que el uso de drones como herramienta de apoyo en la logística trae beneficios significativos. Esta tesis aborda el problema flexible del vendedor viajero con múltiples drones (flexible drones traveling salesman problem, FDTSP por sus siglas en inglés). Este problema busca la combinación óptima entre un camión y uno o múltiples drones que minimice el tiempo para realizar entregas. Este tipo de problema se caracteriza por su complejidad computacional, es por ello que los algoritmos presentes en la literatura sobre el FDTSP obtienen soluciones alejadas del óptimo. En el presente estudio se propone una matheurística para abordar el FDTSP. Este enfoque utiliza de manera alternada un modelo de programación lineal entera mixta (mixed integer linear programming, MILP por sus siglas en inglés) y una metaheurística basada en el variable neighborhood search. La metaheurística opera mediante dos procedimientos de optimización. El primer procedimiento enfocado en optimizar la asignación de clientes a los vehículos y la ruta del camión, utilizando tres operadores basados en el problema del vendedor viajero. El segundo procedimiento está centrado en optimizar las rutas de los drones, utilizando una metaheurística de simulated annealing. La matheurística es validada en dos conjuntos de instancias que representan condiciones de operación en entornos urbanos, suburbanos y rurales. En la mayoría de las instancias, los resultados computacionales muestran que la matheurística propuesta supera, en promedio, a los algoritmos de la literatura. Adicionalmente, la matheurística obtiene los mejores rendimientos cuando se utiliza una cantidad pequeña de drones y se evalúan instancias con un mayor número de clientes. Estos resultados son validados mediante un test estadístico, indicando que nuestro enfoque es estadísticamente diferente de los algoritmos de la literatura. Adicionalmente, para instancias pequeñas de hasta diez nodos, es posible comprobar que se alcanza el resultado óptimo con la matheurística y el modelo de MILP.

Keywords – FDTSP, Matheurística, Modelo MIP, Metaheurística

Abstract

Different studies demonstrate that using drones as a support tool in logistics brings significant benefits. This thesis addresses the flexible drones traveling salesman problem (FDTSP). This problem seeks the optimal combination between a truck and one or multiple drones that minimizes delivery time. This type of problem is characterized by its computational complexity, which is why the state-of-the-art algorithms in the FDTSP literature yield far from optimal solutions. In the present study, a matheuristic is proposed to address the FDTSP. This approach alternates between a mixed integer linear programming (MILP) model and a metaheuristic based on variable neighborhood search. The metaheuristic operates through two optimization procedures. The first procedure focuses on optimizing the assignment of customers to vehicles and the truck's route, using three operators based on the proposed traveling salesman problem. The second procedure focuses on optimizing the drones' routes using a simulated annealing metaheuristic. The matheuristic is validated on two sets of instances representing operating conditions in urban, suburban, and rural environments. In most instances, the computational results show that the proposed matheuristic outperforms, on average, the algorithms in the literature. These results are validated through a statistical test indicating that our approach is statistically different from the algorithms in the literature. Additionally, for small instances with up to ten nodes, it is possible to verify that the optimal result is achieved in both the matheuristic and MILP model.

Keywords – FDTSP, Matheuristic, MILP model, Metaheuristic

Contents

List of Tables

List of Figures

Chapter 1

Introduction

This chapter presents a brief motivation and introduction of the work, the hypothesis, the general objective, the specific objectives and the structure of the document.

1.1 Motivation

Thanks to capabilities, potentialities, and the use of information technologies, various applications have emerged for unmanned aerial vehicles (UAV), also known as drones. One of the applications is in the field of agriculture, where drones can be used for areas mapping, classification and vigilance of crops, application of fertilizer or pesticides, and detection of weeds or crops affected by pests (Rejeb et al., 2022). Moreover, UAVs can be utilized in emergency contexts, proving invaluable for inspection and rescue operations in inaccessible or large areas (Mohd Daud et al., 2022). Another application is in the healthcare field, drones are beneficial for the provision of medical services, facilitating the shipment of medicine and medical supplies (Roberts et al., 2023). Furthermore, drones can perform in the logistics field, effectively addressing the so-called *last mile* problem, allowing the reduction of operation costs, delivery times and emissions reduction (Madani & Ndiaye, 2022).

In logistics field, companies related to delivery services have significantly increased the testing of UAVs in response to the growing demand for online commerce-related services, with the intention of providing a better and faster service. Recently, companies like Amazon and Walmart have increased the volume of services provided, accompanied by

the approval of the Federal Aviation Administration (Amazon, 2024, Faithfull, 2024). Meanwhile, technology companies dedicated to parcel delivery with drones continue to increase their participation (Pope, 2024, Rogers, 2024). Moreover, the use of drones has reached extraordinary altitudes, with the first successful oxygen and supplies delivery on Mount Everest having been completed (DJI, 2024).

While in academia, the last-mile problem has also been addressed through the use of drones. Several authors have proposed the combined use of these UAV with delivery trucks. In this way, researchers have expanded the formulation of the well-known traveling salesman problem (TSP). Murray & Chu (2015) conducted one of the earliest works that considers the use of a truck and a drone, calling the approach as the flying sidekick TSP (FSTSP). Subsequently, Agatz et al. (2018) followed the same logic but changing essential assumptions that allow the approach to be classified differently, calling it the TSP with drone (TSP-D). The main distinguishing assumption of the TSP-D is that drones can visit the same location more than once, permitting loops in their flights. Over the years, diverse authors have approached these problems in diverse ways, establishing assumptions given the defined contexts and using different resolution methods. Furthermore, in the study conducted by Cavani et al. (2021), they expand the TSP-D by considering the use of multiple drones, calling the approach the TSP with multiple drones (TSP-mD). Notably, as discussed by Lu et al. (2022), also motivated by the TSP-D, they expand the problem to include multiple drones, but in this approach they consider the service time of vehicles to customers and the time required to prepare the launch of the drones. Additionally, solution methods are proposed, referring to this approach as the flexible drones traveling salesman problem (FDTSP).

In this thesis, we propose a matheuristic for solving the FDTSP. We employ a metaheuristic and a mixed-integer linear programming (MILP) model to tackle this problem, alternating the use of both methods depending on a given time limit of execution. The proposed metaheuristic approach is characterized by constructing a solution using a two-procedure structure, within an adapted variable neighborhood search (VNS) framework. An initial solution is generated first by solving a TSP problem using the well-known Lin-Kernighan heuristic, followed by constructing routes to serve customers with drones. Meanwhile, the first procedure focuses on improving the service assignment and the truck route, while the second procedure focuses on improving the drone route through a simulated annealing metaheuristic.

1.2 Hypothesis

It is possible to implement a matheuristic that is capable of achieving competitive results compared to the proposal of Lu et al. (2022) for the FDTSP.

1.3 General objective

Develop and implement a matheuristic that obtains competitive results regarding the state-of-the-art algorithms for the FDTSP.

1.4 Specific objectives

- Review the state-of-the-art for the FDTSP and similar problems.
- Design a matheuristic that exploits the advantages of a metaheuristic and a MILP model.
- Implement the proposed matheuristic.
- Analyze and compare the results with the state-of-the-art.

1.5 Document outline

The remaining chapters of this thesis are organized as follows. Chapter 2 presents formally the considered problem and the related literature is reviewed. In Chapter 3, the problem is modeled using a MILP model, describing the proposed constraints and their computational complexity. Then, in Chapter 4, the proposed matheuristic is introduced, providing a detailed explanation of combined use of the metaheuristic and the MILP model. Chapter 5 shows computational experiments and results are analyzed. Finally, Chapter 6 concludes the work.

Chapter 2

Flexible drones traveling salesman problem

This chapter formally describes the considered problem and presents a revision of related problems.

2.1 Problem description

The FDTSP was designed by Lu et al. (2022) to address the problem through route assignment for a single truck and one or multiple drones. Drones are allowed to be launched and retrieved from any location corresponding to a customer served by the truck. Thus, drones may or may not return to the same launch location to be received. The truck is equipped with a specified number of drones and has no load or route limitations. Given the assumption that the transported packages have an insignificant weight relative to the drone's capacity, a drone has no weight limit, can only carry one package at a time, and can make trips based on battery capacity. Shipments can be delivered by either the truck or the drones. The objective is to minimize the truck's return time to the depot after all customers have been served.

The problem can be described with a directed complete graph $G = (V, A)$. The set of nodes is $V = \{0, 1, ..., n-1, n\}$ where 0 and n represent a unique depot, and $N = \{1, ..., n-1\}$ represents the set of customers to be served. Additionally, $N^0 = N \cup \{0\}$ and $N^n = N \cup \{n\}$ correspond to the set of customers to be served along with the

depot and its duplicate, respectively. Meanwhile, the set of edges $A = \{(0, j) : j \in N\} \cup \{(i, j) : i, j \in N, i \neq j\} \cup \{(i, n) : i \in N\}$ indicates all possible travels that a vehicle can make from a node to another. Each drone trip may be represented by $L = \{(i, j, k) : i \in N^0, j \in N, k \in N^n, i \neq j \wedge k \neq j\} \setminus \{(0, j, n) : j \in N\}$. Moreover, the truck starts the route at depot 0 and ends at depot n. This land vehicle can transport identical D drones, each of which serves one customer per trip.

Additionally, vehicles have a travel time calculated based on the speed at which they travel, in Manhattan distance for the truck and in Euclidean distance for drones. Thus, the travel times for each arc are represented by t^t_{ij} y t^d_{ij}, $(i, j) \in A$, for the truck and drones, respectively. Each customer has a service time depending on whether it is served by the truck (s^t) or a drone (s^d). It is important to note that a customer can be served by only one vehicle, and in situations where multiple vehicles converge at a node, it is considered that the truck carries out the service. Moreover, the launch of a drone has an associated set-up time (t^s), and each drone cannot undertake a trip longer than the limited time provided by its battery (ε), considering distances traveled, service time, and waiting time for the truck arrive. It is assumed that before each launch, the drone has a full battery. When a drone is launched from a node and returns to the same launching place, the truck must remain at the location until the drone returns. Meanwhile, if a vehicle arrives at a customer location before another vehicle arrival, it must remain in place and wait until the other vehicle arrives. In the event that a drone needs to wait, it remain in flight consuming its battery until the truck arrives.

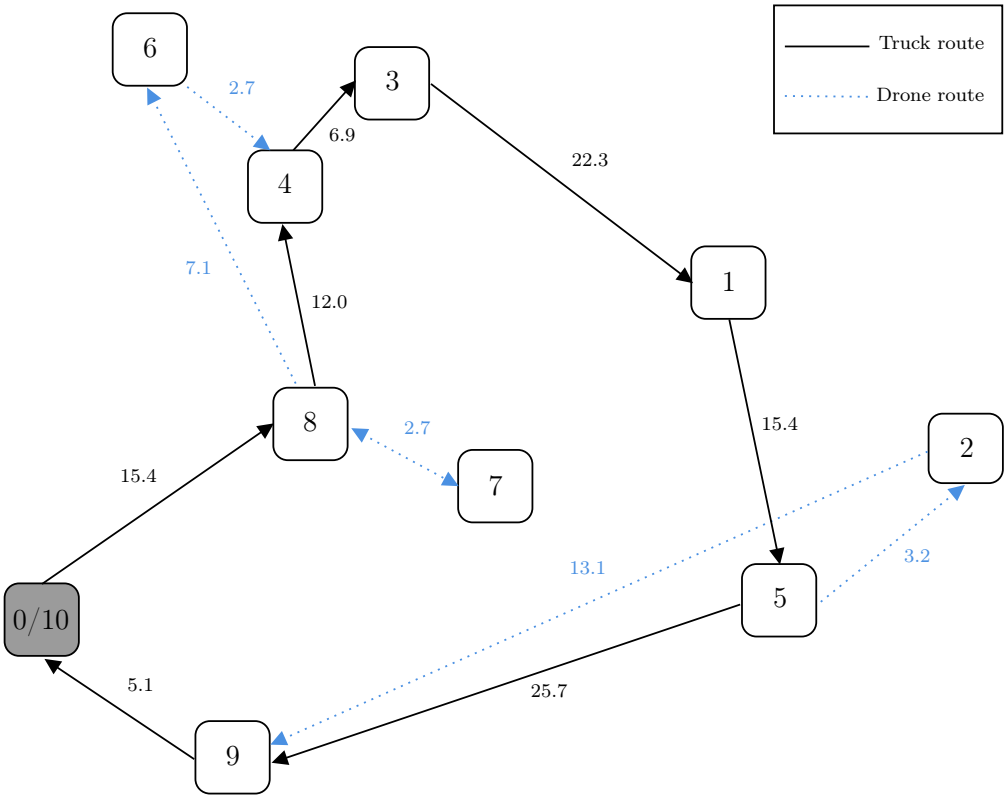

Figure 1: An example of a feasible solution for nine customers and two drones.

Table 1: Arrival times at each node.

Node	0 (10)	8	7	6	4	3	1	5	2	9	10 (0)
Arrival time [min]	0	15.4	20.6	25	35.8	43.2	66	81.9	86.6	109.1	114.7

Figure 1 illustrates an example of a feasible solution for the problem with nine customers and two available drones. The solid black lines represent the truck's route, while the dashed blue lines correspond to the drones' routes. The values associated with the edges represent the time it takes for the vehicles to travel from one node to another. In this example, the drones have a battery life of 60 minutes and a speed of 80 mph, while the truck has a speed of 35 mph. Due to this, the vehicles require the indicated time in the figure to travel over the edges. The launch of a drone takes place after the truck has served the customer and once each drone has been prepared for launch. In this case, a single drone requires 1 minute for its launch, and each customer requires 0.5 minutes to be served by any vehicle. We can see the truck and its drones depart from the depot and head to customer 8. Then, two drones are launched simultaneously from customer location 8, one heading towards customer 6 and the other towards customer 7. The drone

serving customer 7 returns to the same launch node, so the truck must wait for its arrival before moving to customer location 4 to serve it and meet the other launched drone. The drone that served customer 6 must wait for the truck's arrival at the location of customer 4. The truck continues its route, serving customers 3, 1, and 5. Then, it launches another drone at node 5, which serves customer 2. Afterward, this drone heads to the location of customer 9, where it waits for the truck's arrival. Finally, the truck arrives at customer 9's location, serves this customer, and heads toward the depot, concluding the route in 114.7 minutes. Table 1 shows the rounded arrival times at the nodes for the previously described example.

2.2 Literature review

There are several approaches to the logistics operation involving a combination of a truck and drones. A system can be composed of one or multiple trucks that carry out deliveries in coordination with one or multiple drones. Additionally, as Garg et al. (2023) state, there are systems where the operation of drones is independent of the truck, called pure delivery (PM). There are also operations where delivery vehicles work together, called synchronized delivery (SM), offering benefits like increased ranges, cost savings and energy efficiency.

The problems presented in this review of the state-of-the-art belong to the SM classification, considering the use of a single truck. Additionally, the review is restricted to problems that do not consider dynamic assumptions because the main reference study does not consider them. These assumptions can refer to customers' serving that may be generated while vehicles move on the route, as addressed by Zhang & Woensel (2023). Furthermore, the focus is on studies inspired by two main problems, the FSTSP and the TSP-D, along with their variants that propose using multiple drones, which will be explored in detail below.

One of the first studies that proposed the combination of a truck and one drone was conducted by Murray & Chu (2015). One of the problems they addressed is the FSTSP, which involves a single delivery truck traveling a route of assigned customers, while a drone is launched at a customer location visited by the truck. The drone serves another individual customers and meets with the truck on the road. The researchers implement a MILP model and a heuristic, demonstrating the effectiveness of the combined use of the vehicles for making deliveries for up to 10 customer.

In another approach to solve the FSTSP, Ferrandez et al. (2016) designed an algorithm which use a k-means clustering algorithm to find launch locations and a genetic algorithm (GA) to determine truck route between those launch locations. Experimenting with a number of customers ranging from 50 to 250, the study found improvements and state that multiple drones per truck are more efficient and contribute to savings in both energy and time.

Then, Ha et al. (2018) take the FSTSP proposed by Murray & Chu (2015) and adapt it to minimize the operational cost of the route instead of time. These authors propose a MILP model, a heuristic, and a greedy randomized adaptive search procedure (GRASP) to solve the adaptation. Computational results show that GRASP outperforms the heuristic in instances with sizes ranging from 10 to 100 nodes, while the model can find optimal solutions for instances with only 10 nodes.

Dell'Amico et al. (2021) address the FSTSP by a branch and bound algorithm (B&B) for small instances and a heuristic algorithm with B&B as a subroutine to deal with larger instances. These two methods are effective over instances with 20 to 229 customers. Then, the same authors, Dell'Amico et al. (2022) proposed exact models for the FSTSP. Computational results show optimally for small and medium instances, raging from 10 to 20 customers.

Lately, Schaumann et al. (2024) expand the FSTSP, calling it the FSTSP with multiple drops, where the drone can be launched from the truck to deliver multiple packages before it is retrieved. They propose an algorithm based on a local search and diversification techniques. They evaluate their heuristic obtaining better solutions than the state-of-the-art algorithm, considering instances ranging from 50 to 250 customers.

Next changing perspective, but continuing to use a truck and a drone, Agatz et al. (2018) define the TSP-D. In general aspects, the TSP-D is very similar to the FSTSP. However, unlike the FSTSP proposed by Murray & Chu (2015), the TSP-D has a drone that can join the truck at the node where it was released, and the drone is faster than the truck. Due to the computational complexity to address the TSP-D, they propose an integer linear programming model and heuristic algorithms based on local search and dynamic programming (DP). The results demonstrate substantial savings with the TSP-D approach compared to the traditional TSP. Additionally, it is shown that the model optimally solves instances with up to 12 nodes, while the heuristics are evaluated with instances ranging in

size from 10 to 100 nodes. Likewise, Bouman et al. (2018) also addressed the TSP-D and successfully solved larger instances using DP approaches.

Schermer et al. (2020) address the TSP-D with exact approaches. They designed MILP models and a branch and cut (B&C) algorithm that optimally solves instances with a size between 9 to 20 customers. Meanwhile, Nguyen et al. (2020) address the same problem through a heuristic method based on monte carlo tree search. Their proposal outperforms the reviewed literature on instances from 52 to 200 nodes.

Roberti & Ruthmair (2021) propose a compact MILP model and a DP algorithm combined with an exact approach based on branch and price for solving the TSP-D and other variants. Unlike other reviewed problems, the problem considered do not consider travel constraints for drones, and customers do not have associated service times. They tackle instances ranging in size from 10 to 40 customers. The model and the DP approaches yields competitive results concerning to the state-of-the-art algorithms.

Based on the study by Agatz et al. (2018), AlMuhaideb et al. (2021) propose a GRASP metaheuristic that utilizes two local searches and an adaptive selection of the neighborhood for solving the TSP-D. They obtain results better or similar to the state-of-the-art algorithms, considering instances ranging from 10 to 100 nodes. In the same year of publication as the two previous studies reviewed, Vásquez et al. (2021) dealt with the TSP-D, designing a Benders decomposition algorithm. They achieved effectiveness solving instances with 10 to 20 customers.

Meanwhile, El-Adle et al. (2023) proposed a MILP model and a two-phased VNS combined with an exact approach. The heuristic approach consists of exploring the feasible space and a featuring intensification procedure, which the exact approach is used as a re-optimization procedure. They also allow the launch of the same drone more than once from a launching node, which they call a multiple cycle. Computational experiments over instances from 10 up to 100 nodes show improvements in delivery times compared to the literature.

One last published study related to the TSP-D is the approach by Tirkolaee et al. (2024), who expanded drone use by adding a bicycle as a complementary delivery vehicle. This problem was called the TSP-D and bicycle, and they proposed a MILP model to address it. Their approach achieved time savings for instances with up to 50 customers, compared to the reviewed literature.

Regarding the literature on the use of multiple drones and a truck, we first focus on studies related to the FSTSP. Tu et al. (2018) extend the FSTSP proposed by Ha et al. (2018) with the consideration of using multiple drones and a single truck, known as the multiple FSTSP (m-FSTSP). The researchers adapt the GRASP proposed by Ha et al. (2018), in addition to introducing an adaptive large neighborhood search (ALNS). They demonstrate that both approaches find efficient solutions for instances with 51 to 101 nodes, while the ALNS consistently producing better solutions.

In the research conducted by Chang & Lee (2018), like Ferrandez et al. (2016), the authors proposed a k-means algorithm followed by the employment of a nonlinear programming model to shift the centers of clusters and route the truck path. This approach show better results than the approach considering clustering without shifting the centers and a simple routing of the vehicles, evaluating over instances with 10 to 100 nodes.

Then, Mahmoudi & Eshghi (2022) extend the m-FSTSP setting that certain customers can only be served by drones. Their approach allows for launch and retrieval operations non-customer locations, including customer nodes and the depot. Furthermore, drones can perform multiple visits per trip and this UAVs have battery and payload maximum capacity. They called the problem as the energy-constrained multi-visit TSP with multiple drones considering non-customer rendezvous locations. The authors proposed a MILP model and a matheuristic. The second approach is characterized by having several phases, utilizing heuristics for customer assignment, improvement and diversification of a solution, meanwhile the MILP model is used for drone routing. Computational results show that the use of multiple drones leads to a reduction in the makespan over instances with 8 to 50 customers.

Salama & Srinivas (2022) extend the m-FSTSP delivery problem, without considering drone flights with cycles. They called the problem collaborative truck–drone routing and scheduling problem with flexible launch and recovery locations. They include an additional set of truck stop, where the UAV can be launch or retrieve. To solve the approach, the authors proposed a two-phase search algorithm, which consists in three parts. The first determines the assignment of nodes for each vehicle and the truck route, using a simulated annealing (SA) and a VNS. Then, the second part distributes the drones to each customer by a heuristic. Finally, the last part optimizes the scheduling of the vehicles using a MILP model. Computational results show improvements, for instances with 8 to 50 customers, compared to the collaborative truck–drone routing and scheduling problem with restricted

launch and recovery locations.

Conversely, Rinaldi et al. (2023) directly address the m-FSTSP by a MILP model formulation and a local search algorithm, two hybrid-GAs , and a greedy algorithm. They obtain the optimal schedules that result in significant savings in delivery time. The algorithms were evaluated over realistic scenarios, considering 10 and 20 customers to be served.

Regarding the TSP-D with multiple drones. Cavani et al. (2021), building upon the TSP-D approach proposed by Bouman et al. (2018), designed a variant with multiple drones (TSP-mD). They proposed a compact MILP model and addressed it through an exact decomposition approach with a B&C algorithm. They demonstrate that the model is effective in optimally solving instances of up to 25 nodes in less than two hours of computing time, surpassing other exact methods mentioned in the literature that can only handle instances of up to 10 nodes.

Later, based on the TSP-D, Lu et al. (2022) proposed the FDTSP that considers multiple drones capable of flying to and from the same launch node. Therefore, the FDTSP is analogous to the TSP-mD, since the aim is to minimize travel time. The name of the FDTSP refers to their proposed methodology. They propose a clustering algorithm to assign a certain number of customers to be served by drones and the truck. A metaheuristic is used to optimize the truck's route, they propose two metaheuristics to minimize the truck's travel time, a SA and a GA, and the use of Google's OR-tools solver. Additionally, they formulate a MILP model. The results of using metaheuristics demonstrate significant reductions in operation times when compared to a route executed solely by a truck, i.e., a TSP, considering instances ranging from 15 to 150 nodes.

Then, Tiniç et al. (2023) address a variant TSP-mD, aiming to minimize the operational cost of routes associated with a single truck and multiple drones. This study differs from others in that drones are allowed to return to their launch location to be retrieved by the truck. Additionally, the study compares the objective functions of minimizing operational cost and arrival time. To solve the problem, they propose a MILP model and a B&C algorithm. The results outperform other exact approaches for instances with 20 nodes, demonstrating that minimizing arrival time produces better solutions than seeking to minimize operational cost.

Cai & Qian (2023) designed a MILP model and a hybrid-VNS, which has several shaking

operators for improving the solution. They demonstrated that their approach is effective on instances ranging 8 to 100 customers. Recently, Yılmaz et al. (2024) proposed a fitness-distance balance-based evolutionary search algorithm. This proposed algorithm is based on search phases, where a greedy algorithm is used to select a strategy of exploration and intensification of a solution. The experiments conducted on instances of 30 to 100 customers, outperforming the compared algorithms.

Table 2 provides a summary of the mentioned literature. "Category" column details the problem group to which the reviewed problem belongs. "Article" column contains the reference to the reviewed study, "Objective" is the study's sought-after result, aiming to minimize either time (t) or operational cost (o). "Exact method" column refers to the model formulation (✓) and/or an exact approach to solve the model formulation (✓✓). "Heuristic" column corresponds to the heuristic approach used to solve the problem (✓✓). "Matheuristic" column indicates that the problem was solved using a combination of an exact method and an approximate one (✓✓). Finally, the "max $|V|$" column indicates the maximum number of nodes present in each problem.

Table 2: Summary of literature review.

Category	Article	Objective	Exact method	max $\|V\|$	Heuristic	max $\|V\|$	Matheuristic	max $\|V\|$
FSTSP	Murray & Chu (2015)	t	✓✓	11	✓✓	21	-	-
	Ferrandez et al. (2016)	t	-	-	✓✓	251	-	-
	Ha et al. (2018)	o	✓✓	11	✓✓	101	-	-
	Dell'Amico et al. (2021)	t	✓✓	16	-	-	✓✓	230
	Dell'Amico et al. (2022)	t	✓✓	21	-	-	-	-
	Schaumann et al. (2024)	t	-	-	✓✓	251	-	-
TSP-D	Agatz et al. (2018)	t	✓✓	12	✓✓	100	✓✓	100
	Bouman et al. (2018)	t	✓✓	21	-	-	-	-
	Schermer et al. (2020)	t	✓✓	21	-	-	-	-
	Nguyen et al. (2020)	t	-	-	✓✓	200	-	-
	Roberti & Ruthmair (2021)	t	✓✓	40	-	-	-	-
	AlMuhaideb et al. (2021)	t	-	-	✓✓	100	-	-
	Vásquez et al. (2021)	t	✓✓	21	-	-	-	-
	El-Adle et al. (2023)	t	✓	-	-	-	✓✓	100
	Tirkolaee et al. (2024)	t	✓✓	51	-	-	-	-
m-FSTSP	Tu et al. (2018)	o	✓	-	✓✓	101	-	-
	Chang & Lee (2018)	t	✓	-	-	-	✓✓	101
	Mahmoudi & Eshghi (2022)	t	✓✓	11	-	-	✓✓	51
	Salama & Srinivas (2022)	t	✓✓	9	-	-	✓✓	51
	Rinaldi et al. (2023)	t	✓	-	✓✓	21	-	-
	Cavani et al. (2021)	t	✓✓	25	-	-	-	-
TSP-mD	Lu et al. (2022)	t	✓	-	✓✓	150	-	-
	Tiniç et al. (2023)	o	✓✓	20	-	-	-	-
	Cai & Qian (2023)	t	✓✓	11	✓✓	101	-	-
	Yılmaz et al. (2024)	t	✓	-	✓✓	101	-	-
	This work	t	✓✓	10	-	-	✓✓	150

Chapter 3

Mixed integer linear programming model

This chapter presents the proposed MILP model and describes the set of parameters, variables, and constraints associated with the model proposed.

3.1 MILP model

Lu et al. (2022) formulated a mathematical model that is not suitable for implementation in a general-purpose solver, as it either overlooks or only partially addresses key operational constraints and practical considerations, such as drones flights balancing or battery endurance, among others. Therefore, we propose a MILP model, and given the similarities between the TSP-mD and the FDTSP, our model is based on the work of Cavani et al. (2021), incorporating the necessary assumptions to ensure that all customers are served while minimizing the travel time of the delivery truck in coordination with the drones.

The MILP model uses the binary decision variable x_{ij}, $(i, j) \in A$, which takes the value 1 if the truck travels from node i to j. Additionally, the binary variable y_i takes the value 1 if the truck serves customer $i \in N$. Likewise, the binary variable z_{ijk}, $(i, j, k) \in L$, takes the value 1 if the drone is launched from node i, serves customer j, and is received at node k. Moreover, an integer variable is defined, w_i representing the number of drones flying at the moment the truck leaves node $i \in N$. Finally, the continuous variable a_i represents the arrival time of a vehicle at node $i \in N$. We summarize the sets, the parameters, and

decision variables used for the proposal MILP model and list them in Table 3.

Table 3: Summary of sets, parameters, and decision variables.

Sets	Description
V	Node set $\{0,1,2,\ldots,n-1,n\}$.
N	Customer set $\{1,2,\ldots,n-1\}$.
N^0	Set of customers and the depot $\{0,1,2,\ldots,n-1\}$.
N^n	Set of customers and the duplicate depot $\{1,2,\ldots,n-1,n\}$.
A	Set of edges.
L	Set of nodes visited by a drone on possible flights.
Parameters	**Description**
M	Very large number.
D	Initial number of drones available.
ε	Time in minutes of a drone's battery life.
t^s	Setup time of a drone.
s_i^t	Time the truck spends servicing customer $i \in N$.
s_i^d	Time a drone spends servicing customer $i \in N$.
t_{ij}^t	Time the truck takes to travel from node i to j, $\forall (i,j) \in A$.
t_{ij}^d	Time a drone takes to travel from node i to j, $\forall (i,j) \in A$.
Decision Variables	**Description**
x_{ij}	Binary decision variables representing the truck's trip from node i to j, $(i,j) \in A$.
y_i	Binary decision variables that indicate that customer $I \in N$ is served by the truck.
z_{ijk}	Binary decision variables indicating the trip of a drone launched from customer i, serves customer j, and returns to k, $(i,j,k) \in L$.
w_i	Integer decision variables denoting the number of drones flying at the moment the truck leaves customer $I \in V$.
a_i	Continuous decision variables representing the time at which a vehicle arrives at node $i \in V$.

The following MILP model formulation is proposed:

$$\text{minimize } a_n \tag{1}$$

subject to:

$$\sum_{j\in N^0:j\neq i} x_{ji} = \sum_{j\in N^n:j\neq i} x_{ij} \qquad \forall i \in N \tag{2}$$

$$\sum_{j\in N^n:j\neq i} x_{ij} = y_i \qquad \forall i \in N \tag{3}$$

$$\sum_{j\in N} x_{0j} = \sum_{i\in N} x_{in} = 1 \tag{4}$$

$$y_j + \sum_{i\in N^0:i\neq j} \sum_{k\in N:k\neq j\neq i} z_{ijk} + \sum_{i\in N:i\neq j} z_{ijn} + \sum_{i\in N:i\neq j} z_{iji} = 1 \qquad \forall j \in N \tag{5}$$

$$y_i D \geq \sum_{j\in N:j\neq i} \sum_{k\in N^n:k\neq j\neq i} z_{ijk} + \sum_{j\in N:j\neq i} z_{iji} \qquad \forall i \in N \tag{6}$$

$$D \geq \sum_{j\in N} \sum_{k\in N:k\neq j} z_{0jk} \tag{7}$$

$$M y_i \geq \sum_{j\in N:j\neq i} \sum_{k\in N^n:k\neq j} z_{ijk} + \sum_{k\in N^0} \sum_{j\in N:j\neq k\neq i} z_{kji} \qquad \forall i \in N \tag{8}$$

$$w_o = \sum_{j\in N} \sum_{k\in N:k\neq j} z_{0jk} \tag{9}$$

$$w_n = 0 \tag{10}$$

$$w_o \leq D \sum_{j\in N} x_{0j} \tag{11}$$

$$w_i \leq D \sum_{j\in N^n:j\neq i} x_{ij} \qquad \forall i \in N \tag{12}$$

$$w_0 - \sum_{r\in N:r\neq j} z_{0rj} + \sum_{r\in N:j\neq r} \sum_{s\in N^n:s\neq r\neq j} z_{jrs} \leq w_j + M(1 - x_{0j}) \qquad \forall j \in N \tag{13}$$

$$w_i - \sum_{s\in N^0:s\neq r} \sum_{r\in N:r\neq s\neq j} z_{srj} + \sum_{r\in N:j\neq r} \sum_{s\in N^n:s\neq r\neq j} z_{jrs} \leq w_j + M(1 - x_{ij}) \quad \forall i,j \in N : j \neq i \tag{14}$$

$$w_i - \sum_{s\in N} \sum_{r\in N:r\neq s} z_{srn} + \leq w_n + M(1 - x_{in}) \qquad \forall i \in N \tag{15}$$

$$\sum_{r\in N:r\neq j}\sum_{s\in N:s\neq r} z_{jrs} \leq D - w_i + \sum_{s\in N^0:s\neq j}\sum_{r\in N:r\neq s\neq j} z_{srj}$$
$$+ M(1-x_{ij}) \quad \forall i \in N^0, j \in N : i \neq j \tag{16}$$

$$a_0 = 0 \tag{17}$$

$$a_0 + (M + t^t_{0j})x_{0j} + \sum_{r\in N}\sum_{s\in N:s\neq r} t^s z_{0rs} \leq a_j \qquad \forall j \in N \tag{18}$$

$$a_0 + \sum_{k\in N:k\neq j} (M + t^d_{0j})z_{0jk} + \sum_{r\in N}\sum_{s\in N:s\neq r} t^s z_{0rs}$$
$$\leq a_j + M \qquad \forall j \in N \tag{19}$$

$$a_0 + \sum_{r\in N}\sum_{s\in N:s\neq r} t^s z_{0rs} + (M + t^d_{0l} + s^d_l + t^d_{lj})z_{0lj}$$
$$\leq a_j + M \quad \forall j,l \in N : l \neq j \tag{20}$$

$$a_i + (M + t^t_{ij} + s^t_i)x_{ij} + \sum_{r\in N}\sum_{s\in N^n:s\neq r} t^s z_{irs}$$
$$+ (t^d_{il} + s^d_l + t^d_{li})z_{ili} \leq a_j + M \quad \forall i,j,l \in N : i \neq j \neq l \tag{21}$$

$$a_i + \sum_{r\in N}\sum_{s\in N^n:s\neq r} t^s z_{irs} + \sum_{k\in N^n:k\neq j} (M + s^t_i + t^d_{ij})z_{ijk}$$
$$\leq a_j + M \quad \forall i,j \in N : j \neq i \tag{22}$$

$$a_i + s^t_i + \sum_{r\in N}\sum_{s\in N^n:s\neq r} t^s z_{irs} + (M + t^d_{il} + s^d_l + t^d_{lj})z_{ilj}$$
$$\leq a_j + M \quad \forall i,j,l \in N : i \neq j \neq l \tag{23}$$

$$a_i + (M + t^t_{in} + s^t_i)x_{in} + \sum_{r\in N}\sum_{s\in N^n:s\neq r} t^s z_{irs}$$
$$+ (t^d_{ij} + s^d_j + t^d_{ji})z_{iji} \leq a_n + M \quad \forall i,j \in N : j \neq i \tag{24}$$

$$a_k - a_0 - \sum_{r\in N}\sum_{s\in N:s\neq r} t^s z_{0rs}$$
$$- M(1 - z_{0,j,k}) \leq \varepsilon \quad \forall j,k \in N : k \neq j \tag{25}$$

$$a_k - a_i - \sum_{r\in N}\sum_{s\in N^n:s\neq r} t^s z_{irs} - s^t_i - M(1 - z_{ijk})$$
$$\leq \varepsilon \quad \forall i,j,k \in N : i \neq j \neq k \tag{26}$$

$$t^d_{ij} + s^d_j + t^d_{ji} - M(1 - z_{iji}) \leq \varepsilon \quad \forall i,j \in N : j \neq i \tag{27}$$

$$t^d_{ij} + s^d_j + t^d_{jn} - M(1 - z_{ijn}) \leq \varepsilon \quad \forall i,j \in N : j \neq i \tag{28}$$

$$x_{ij} \in \{0,1\} \qquad \forall (i,j) \in A \tag{29}$$

$$y_i \in \{0,1\} \qquad \forall i \in N \tag{30}$$

$$z_{ijk} \in \{0,1\} \qquad \forall (i,j,k) \in L \qquad (31)$$

$$w_i \in \mathbb{N}^0 \qquad \forall i \in V \qquad (32)$$

$$a_i \in \mathbb{R}^+ \qquad \forall i \in V \qquad (33)$$

The objective function (1) aims to minimize the truck's arrival time at the depot. Then, constraints (2) and (3) ensure that the ingoing and outgoing flow of the truck for a customer are the same, and that the customer visited is served by this vehicle. Constraints (4) state that the truck must depart from and return to the depot. Constraints (5) limit that each customer must be served by only one vehicle. Constraints (6) and (7) limit the number of launches to the maximum number of available drones from a node or from the depot. Constraints (8) ensure that a drone can only be launched or received at a node visited by the truck. Constraints (9) and (10) guarantee the number of drones flying from and to the depot. Meanwhile, constraints (11) and (12) limit the flights from and to nodes according to the truck's presence. Constraints (13)–(16) balance the number of drones flying according to launches and receptions at each node. The following are the arrival time constraints, where the arrival time at a node visited by both the truck and drones is determined by the truck's recorded time, while the arrival time at a node visited only by a drone is determined by the drone's recorded time. Constraint (17) sets the start time at the depot to zero, allowing constraints (18)–(20) to establish the travel times of the truck and drones starting from the depot. Specifically, constraints (18) define the truck's arrival time at node j. Constraints (19) set the arrival time of a drone at node j provided the drone served the node, while constraints (20) establish the arrival time of a drone at node j if the node is the retrieval point. Constraints (21)–(23), similarly to the previous constraints, set the times considering that travels occur between nodes corresponding to customers. Constraints (24) limit the arrival time from customers to the final depot. Constraints (25) and (26) limit the flights of drones according to the duration of their batteries. Then, constraints (27) and (28) limit the flight of drones that are retrieve at the same launch node or at the final depot. Finally, constraints (29)–(30) define the domains of the decision variables.

Additionally, M corresponds to a very large number and is computed as shown in (34). Thus, the computing corresponds to the sum of the maximum time of the truck's time travels and the maximum time travel of the drone's possible trips. It is important to mention that, due to the use of "big-M" constraints, it is possible for the variables

$a_i \in \mathbb{R}^+, \forall i \in V$ to take values equal to zero, implying that the continuous relaxation of the model is weak (Cavani et al., 2021). However, the valid inequalities proposed by the authors were not included, as they could increase the computing time required by the general-purpose solver to load the model's constraints in the subsequent experimentation.

$$\max_{i,j \in V: i \neq j} \{t^t_{ij}\} + \max_{i,j \in V: i \neq j} \{t^d_{ij}\} \tag{34}$$

Regarding the computational complexity of the proposed MILP model. This model considers $|V|$ continuous variables (a), $|V|$ integer variables (w), and $|V|^3 + |V|^2 + |V|$ binary variables (z, x, and y, respectively). On the other hand, the set of constraints has different computational complexities as shown in the summary in Table 4. The constraints related to truck routing exhibit cubic complexity in the worst case scenario, while those balancing the number of drones in flight have fourth-degree complexity. Additionally, the arrival time constraints present fifth-degree complexity in the worst case.

Finally, the overall complexity of the constraints is $\mathcal{O}(|N^5|)$.

Table 4: Summary of the computational complexity of constraints.

Computational complexity	**Constraints**
$\mathcal{O}(1)$	(1),(10),(17)
$\mathcal{O}(\|N\|)$	(4),(11)
$\mathcal{O}(\|N\|^2)$	(2),(3),(7),(9),(12),(27),(28)
$\mathcal{O}(\|N\|^3)$	(5),(6),(8),(13),(15),(18),(19)
$\mathcal{O}(\|N\|^4)$	(14),(16),(20),(22),(24),(25)
$\mathcal{O}(\|N\|^5)$	(21),(23),(26)
Final complexity	$\mathcal{O}(\|N\|^5)$

Chapter 4

Matheuristic

This chapter presents the proposed algorithm for solving the problem. Following the structure of a matheuristic (MH), a metaheuristic is used in colaboration with an MILP model. Additionally, the shaking operators and neighborhood structures are described.

4.1 General structure of the matheuristic

A MH is a hybrid optimization method that combines exact methods and metaheuristic methods to exploit the complementary strengths of both (Zhang et al., 2024). We employed a MH approach to address the problem in this study. Thus, the utilization between an adapted VNS algorithm and a reduced MILP model is presented in this section.

Our proposed approach, detailed in Algorithm 1, has a general time limit (T^{MH}), a time limit for the reduced model (T^{M}), and a time limit for the VNS (T^{V}). Thus, the VNS and the MILP model are executed iteratively depending on the remaining time. Within the MH framework, the VNS is used to obtain an initial solution, which is also the best know solution at the moment (line 1). Subsequently, the algorithm's main loop starts (lines 2-9), and depending on the available computing time and the time limit, the reduced MILP (line 3) and the VNS (line 6) iteratively attempt to improve the solution. We consider a reduced MILP model, in order to reduce the computational complexity involved in the MILP model, where a pre-computed solution is used as a warm-start solution. Therefore, some variables and constraints are not considered, achieving the reduction of the search space. Regarding the VNS, this type of algorithm explores different neighborhoods of

a solution. In each iteration, shakings and local searches are applied to explore various neighborhoods and escape local optima. We proposed three neighborhood procedures and four shaking procedures.

Algorithm 1: Matheuristic (MH)

```
1 s̄ ⟵ VNS(T^V)
2 repeat
3 |   s ⟵ reduced-MILP(s̄, T^M)
4 |   if f(s) < f(s̄) then
5 |   |   s̄ ⟵ s
6 |   s ⟵ VNS(s̄, T^V)
7 |   if f(s) < f(s̄) then
8 |   |   s̄ ⟵ s
9 until T^MH is exceeded
```

4.2 Representation of a solution

A solution is represented as a tuple, $s = \{s^T, s^D\}$. s^T is a list containing nodes visited exclusively by the truck in chronological order, representing a Hamiltonian route, with the depot as the first and last element in the list. Meanwhile, s^D, representing the routes taken by the drones, is a list that stores the routes of the drones following a triplet: $\{\pi^1, \pi^2, \pi^3\}$, where π^1 and $\pi^1 3$ represent the launch and reception nodes ($\pi^1, \pi^3 \in s^T$), which must be visited by the truck, and π^2 is the node of a customer served by the drone ($\pi^2 \in s^D \backslash \{\pi^1, \pi^3\}$).

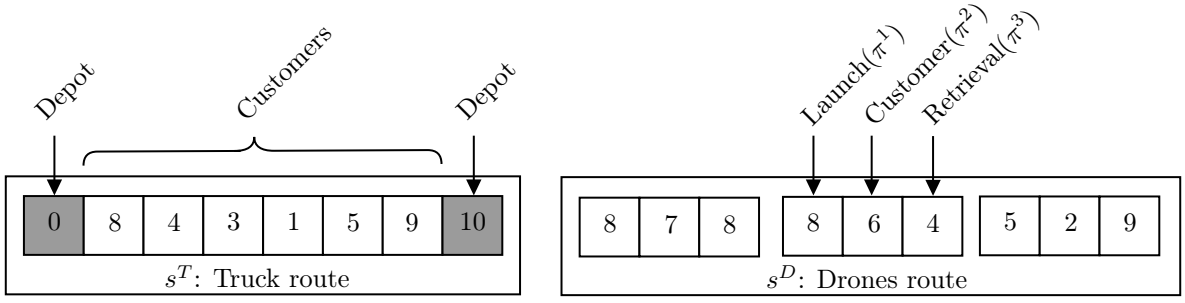

Figure 2: Representation of a solution for nine clients and two drones.

Figure 2 illustrates the proposed representation for a feasible solution. This depicted solution is for the problem with nine customers and two drones shown in Figure 1. The element s^T of the solution contains the nodes visited sequentially by the truck. Starting from the depot 0, visiting 6 customers, and returning to the depot duplicate 10. Meanwhile,

element s^D contains three triplets representing drones trips. In each triplet, the launch and reception nodes can be distinguished, which also belong to the list s^T, and the customer served by the drone on the trip.

4.3 Reduced MILP model

Since the current operators proposed for use in the VNS quickly find possible truck routes using the TSP approach, the main complexity lies in finding the optimal combination and coordination of drones routes. Therefore, we propose using the MILP model to determine the optimal drone routes. Due to the computational complexity of the MILP model proposed in Section 3.1, we remove and reduce all constraints and decision variables related to routing from the model.

The model is reduced as follows. The variables related to the truck route become fixed parameters and the related constraints are not considered. Since a pre-computed truck and its drone routes are considered. Meanwhile, the variables related to drones routes are a warm-start solution, acting as an initial solution for the general-purpose solver. Moreover, the nodes served by the truck and drones are known, so the sets V and N are reduced. Additionally, the truck route's arrival times serve as the lower bound for the associated variables. In the best scenario, they remain the same, but, if a drone arrives with a delay the truck must wait and the arrival times could change. Therefore, the MILP model (1–33) is reduced, considering the constraints (5)–(33), while maintaining the use of the objective function (1), and varying only the sets as appropriate for each case. Finally, the time limit (T^M) is set to a limit depending on each instance. Note that this approach may not find another feasible solution within the time limit, which differs from the initial solution provided, so the matheuristic framework will continue executing to the next step.

4.4 Variable neighborhood search

This step is inspired by the operation of a VNS (Mladenović & Hansen, 1997), where the main iterations are executed until a termination criterion is met. Within these iterations, searches are conducted in neighborhoods, accepting the best solutions. In our proposal, we add an extra procedure within the main iterations, and for the sake of simplicity, we refer to this proposed algorithm as VNS.

The proposed VNS operates by constructing an initial feasible solution, if it does not

receive another initial solution as input. This solution is optimized through two procedures to find the best combination between the truck route and multiple drone trips. Each procedure aims to find the best solution applying different operators, which consist of shaking and local searches in the current solution neighborhood. The first procedure, vehicle route optimization (VRO), is focused on exchanging the customer service performed by different vehicles and improving the truck route, seeking the best combination between the truck's service and the drones. Meanwhile, the second procedure is a SA that aims to optimize the drone routes without changing the customers to be served by the UAVs. The proposed algorithm is presented in the pseudocode of Algorithm 2. Due the combination of various procedures within a VNS framework, this algorithm is technically an hybrid algorithm.

VNS begins with a feasible solution ($\bar{s}$) and a time limit (T^V) as inputs. If a feasible solution is not provided ($\bar{s}$ is empty), an initial solution is computed through a heuristic procedure that ensures the construction of a feasible solution (line 2). Then, the proposed algorithm's main loop is executed until the maximum number of iterations (I^V) is reached or T^V is exceeded (lines 4–16). Within the main loop, the first and second optimization procedures are applied at each iteration that is a multiple of the parameters c_1 and c_2, respectively (lines 5 and 11). Specifically, the VRO operator is executed in K^V neighborhoods only when the main iteration is a multiple of parameter c_1 (lines 5–10), while the SA operator is executed if the main iteration is a multiple of either parameter c_1 or c_2. Thus, when the main iteration is a multiple of parameter c_1, the VRO and SA procedures are executed sequentially. The VRO procedure (line 8) reallocates customers to be served by either the truck or drones and restructures each vehicle's route. This procedure uses three operators, which are explained in Section 4.4.3, and their use is associated with an α probability. These operators aim to improve the solution after a shaking step in the solution's neighborhood. If this shaking results in an infeasible solution, another operator may be called to fix it. In the worst case, the outcome of applying these operators is that all customers are served by the truck. On the other hand, if customers are assigned to be visited by any drone, the SA procedure (line 13) is applied as a local search for optimizing the drone routes. If a solution is found that reduces the total travel time for serving all customers, the best-known solution is updated (lines 9–10 and 14–15). Additionally, if a feasible solution cannot be found, the cost of this solution is penalized by a procedure presented in the next section.

Algorithm 2: Variable neighborhood search (`VNS`)

input : $\bar{s}, T^V$
output : s^*

if $\bar{s}$ is empty **then**
  $\bar{s} \longleftarrow$ `initial-solution`()
$s^* \longleftarrow \bar{s}; i \longleftarrow 1$
while $i \leq I^V$ **or** T^V is exceeded **do**
  if $i\%c_1 = 0$ **then**
    $s \longleftarrow \bar{s}$
    for $k \longleftarrow 1$ **to** K^V **do**
      $\bar{s} \longleftarrow$ `VRO`$(s, \bar{s}, \alpha)$
    if $f(\bar{s}) \leq f(s^*)$ **then**
      $s^* \longleftarrow \bar{s}$
  if $i\%c_1 = 0$ **or** $i\%c_2 = 0$ **then**
    if there are customers assigned to the drones **then**
      $s \longleftarrow$ `SA`$(\bar{s})$
      if $f(s) \leq f(s^*)$ **then**
        $s^* \longleftarrow \bar{s} \longleftarrow s$
  $i \longleftarrow i + 1$

4.4.1 Procedure for penalizing infeasible solutions

As mentioned above, a generated solution could be infeasible within the VNS procedures. Specifically, infeasibility can occur within some triplet constructed for the drone route since the sequences used are designed within operators based primarily on randomness. Therefore, a solution containing an infeasible triplet is considered infeasible. Due to the logic of the procedures, three types of infeasibilities can occur in a triplet: i) the number of drones launched exceeds the available quantity at that moment, ii) the number of drones received exceeds the quantity launched previously, and finally, iii) if the duration of the drone trip exceeds its allowed battery life. An infeasible solution is identified by calculating the cost of the operation between the truck and the drones. Since the element s^D of a solution can contain multiple triplets, it could be infeasible for each triplet. The calculation of the cost of an infeasible solution is based on the truck's arrival time at the depot, considering the feasible drone's routes plus the number of infeasibilities present in the solution, multiplied by a parameter cost (P), which is determined by a tuning procedure.

4.4.2 Initial solution procedure

The solution generation initially consists of constructing a truck route that considers all customers nodes. The initial truck route is constructed by solving a TSP for all nodes, using the LKH solver (Helsgaun, 2000), based on the classic heuristic by Lin & Kernighan (1973). LKH is known for finding a good approximation to the optimal route with effective computational cost. Then, depending on the number of available drones, customers served by the truck are randomly selected to be evaluated for potential service by these UAVs. Once a customer is chosen to be served by a drone, each node in the truck route is assessed to determine if it can serve as a launch or retrieval node. For each node evaluation, the feasibility of the triplet is verified. Additionally, it is ensured that no drone routes cause the truck to wait for the drone's arrival. As soon as a feasible flight is found, the iteration stops, and the customer is confirmed to be served by a drone. This process continues until no drone can serve another customer. If no feasible routes are found for the drones, or if all found routes are infeasible, the initially constructed truck route with all nodes will be used.

4.4.3 Vehicle route optimization procedure

VRO procedure is designed to exchange customer service tasks between different vehicles and optimize the truck route, aiming for the best combination of services between the truck and the drones. VRO uses three operators, each of which involves applying a shaking step followed by a local search. It is important to mention that removing a node from a vehicle route list (either s^T or s^D) is considered a shaking of the solution. The first operator is the reduction of the number of customers served by truck (RCT), which aims to reassign a node served by the truck to be served by a drone, adding the node to the s^D list. The second operator is the increased customer number served by the truck (ICNT), which aims to reassign a node to the truck, adding the node to the s^T list. The last operator is the truck route alteration (TRA), which aims to improve the truck route and may involve modifying the drone's routes as well.

Algorithm 3: Vehicle route optimization (VRO)

```
   input  : s, s̄, α
   output : s̄
 1 s ⟵ RCT(s)
 2 while s is infeasible do
 3 |   s ⟵ ICNT(s)
 4 |   if s is feasible then
 5 |   |   break
 6 |   s ⟵ TRA(s)
 7 |   if s is feasible then
 8 |   |   break
 9 if f(s) ≤ s̄ then
10 |   s̄ ⟵ s
11 if random(0, 1) ≤ α then
12 |   s ⟵ ICNT(s)
13 |   while s is infeasible do
14 |   |   s ⟵ TRA(s)
15 |   |   if s is feasible then
16 |   |   |   break
17 |   |   s ⟵ ICNT(s)
18 |   |   if s is feasible then
19 |   |   |   break
20 else
21 |   s ⟵ TRA(s)
22 |   while s is infeasible do
23 |   |   s ⟵ ICNT(s)
24 |   |   if s is feasible then
25 |   |   |   break
26 |   |   s ⟵ TRA(s)
27 |   |   if s is feasible then
28 |   |   |   break
29 if f(s) ≤ f(s̄) then
30 |   s̄ ⟵ s
```

Algorithm 3 details the VRO procedure, which begins with the use of the RCT operator (line 1). If the resulting solution is infeasible, the ICNT and TRA operators are applied until the solution is feasible (lines 2–8). Then, it checks for any cost improvement. If there is an improvement the best known solution is updated (lines 9–10). Next, based on the parameter α, the same above procedure is applied, but starting with either the ICNT operator (lines 11–19) or the TRA operator (lines 20–28). After completing the loop, it checks again for any cost improvement in the solution (lines 29–30). Details of the operators' functionality are presented below.

- RCT: This operator aims to increase the number of customers served by drones. This starts by randomly selecting a node from the truck route to be removed and reassigned to be served by a drone. This selection is limited to nodes that are not designated as launch or retrieval locations. Additionally, the selection is limited to scenarios where, after removing the node, the truck's route time does not increase due to drone setup time or waiting for drone flights. Once a customer node is selected, a local search is conducted to identify both a launch and a retrieval node for the new drone flight, following the same steps used to construct the drone routes in the initial solution. If a feasible drone route cannot be identified during the search, the node is passed to the ICNT operator, which attempts to reintegrate the selected customer back into the truck's route.

- ICNT: The operator aims to insert a new node into the truck's route. This process can begin in two ways. The first occurs when the operator is randomly selected within the VRO (line 12 of Algorithm 3). In this case, a drone flight is randomly removed from the s^D list, and the node previously served by the drone is reassigned to the truck. The second occurs when another operator invokes this one (lines 3, 17, and 23 of Algorithm 3), in which case the nodes selected by the calling operator are assigned to be served by the truck. Once the nodes to be inserted are identified, they are randomly placed into the truck's route, provided that the insertion does not cause the truck to wait for a drone or cause a drone to exceed its battery limit while waiting for the truck. If the insertion would disrupt a drone flight, the TRA operator is called to reorganize the truck's route, incorporating the node selected in the previous step.

- TRA: This procedure aims to optimize the truck's route. It begins by applying a shaking method to the truck's route, which involves randomly swapping the service order of two customers. Next, a local search is performed using the well-known 2-OPT procedure (Croes, 1958). If these changes make the existing drone routes infeasible, implying that the nodes assigned to a drone cannot be served, the operator reconstructs the drone routes based on the rules from the initial solution. If a feasible route cannot be constructed for one or more nodes, the ICNT operator is called to insert these nodes into the truck's route. In the worst-case scenario, the TRA and ICNT operators may call each other repeatedly until all customers are assigned to be served by the truck.

4.4.4 Simulated annealing procedure

Within the VNS, the SA is used as an operator and is responsible for finding the best drone route combinations to serve the customers assigned to these vehicles. This operator is an adaptation of the SA metaheuristic (Kirkpatrick et al., 1983), which is based on an analogy to metallurgical processes. Starting from an initial state, it searches for a better solution based on diversification. Then, it decides to stay in the neighborhood if an improvement is found or to move to another neighborhood based on probability.

Algorithm 4: SA

input : s^D
output : $\bar{s}^D$

1 $\bar{s}^D \longleftarrow s^D$; $i \longleftarrow 1$; $t \longleftarrow t_0$
2 **while** $T \leq t$ **and** $i \leq I^S$ **do**
3 $s^{D*} \longleftarrow s^D$
4 **for** $k \longleftarrow 1$ **to** K^S **do**
5 $s^D \longleftarrow$ `triplet-change`$(s^D, \sigma_1, \sigma_2)$
6 **if** $f(s^D) < f(s^{D*})$ **then**
7 $s^{\bar{D}} \longleftarrow s^D$
8 **if** $f(s^{D*}) < f(\bar{s}^D)$ **then**
9 $\bar{s}^D \longleftarrow s^{D*}$
10 **else if** `random`$(0,1) < e^{-\frac{|f(s^{D*}) - f(s^D)|}{t}}$ **then**
11 $s^{D*} \longleftarrow s^D$
12 $i \longleftarrow i + 1$; $t \longleftarrow \beta t$

The proposed SA is presented in Algorithm 4, starting from a feasible solution. Thus, to optimize the drone routes, we need to know the truck route and the customers assigned to be served by the drones obtained in the previous phases. If no customers are designated to be served by drones, this phase cannot be carried out. Based on this, we attempted to improve the drone routes (s^D) without modifying the truck route. Thus, SA starts initializing the temperature, iteration counter, and current solution copy (line 1). The main loop (lines 2–12) are carried out as the temperature is reduced geometrically until the target temperature is reached or a maximum number of iterations is performed. A diversification is applied (line 5), and it is evaluated whether there is an improvement (lines 6–9) or the solution is accepted by randomly changing the search neighborhood (lines 10–11). Diversifying and exploring different neighborhoods is governed by the parameter K^S.

The diversification process used in the SA procedure is triplet-change, which involves randomly swapping the launch node, the reception node, or both within the drones triplet with a node visited by the truck. The selection of the new nodes to be part of a drone route consists of randomly choosing a node from the truck route that does not belong to any established drone route. The decision to change one or both nodes has a 50% associated probability. This probability is intended to reduce the likelihood of generating infeasibility and to ensure that the neighborhood change is not too abrupt for the solution. Regarding the acceptance of a solution, if the cost of the solution generated by the diversification process is better than the best-known cost, the solution is updated. Otherwise, we use another criterion to accept a solution, specifically the Metropolis criterion (Metropolis et al., 1953), which is determined by evaluating a quotient between the cost difference and the parameter t in an exponential function.

Chapter 5

Computational experiments

This chapter presents the instances used to evaluate the effectiveness of the proposed MILP model and MH. Additionally, a comparison with the algorithms from the literature is performed.

5.1 Description of the instances

We consider two set of instances, a *large set* and a *small set*. The construction of the *large set* of instances is proposed by Lu et al. (2022), simulating conditions of urban, suburban, and rural areas, with the depot and customers randomly distributed. The urban areas (U) consist of a 10×10 unit area, which can have 80, 100, or 150 nodes. The suburban areas (S) correspond to a 20×20 unit area, which can hold 40, 60, or 80 nodes. Finally, the rural areas (R), with an area of 40×40 units, can have 15, 30, or 40 nodes. Meanwhile, we create a *small set* of instances in order to compare the effectiveness of the proposed MILP model. This set consists of the same three conditions as the previously described *large set* of instances, but the number of nodes is reduced, where each area can have 10, 15, or 20 nodes. Table 5 summarizes the *large set* and *small set* of instances, presenting different scenarios defined by their areas and the number of nodes ($|V|$).

Table 5: Summary of the *large set* and *small set* of instances.

large set		
Scenario	Area $[mile^2]$	$\|V\|$
Urban	10×10	{80, 100, 150}
Suburban	20×20	{40, 60, 80}
Rural	40×40	{15, 30, 40}

small set		
Scenario	Area $[mile^2]$	$\|V\|$
Urban	10×10	{10, 15, 20}
Suburban	20×20	{10, 15, 20}
Rural	40×40	{10, 15, 20}

Each scenario is evaluated using 1, 2, 3, 4, or 5 drones, where each set of drones can have speeds of 40, 60, or 80 miles per hour (mph) and a battery duration of 30 or 60 minutes (min). Meanwhile, the truck's speed remains constant at 35 mph. Additionally, serving each customer takes 0.5 minutes, regardless of the vehicle used for delivery. The setup time for launching a drone is 1 minute. Table 6 summarizes these parameters. It is worth noting that, considering all scenarios and parameters, the *large set* consists of 270 instances, and the *small set* consists of 270 instances, for a total of 540 instances.

Table 6: Summary of the parameters for the sets of instances.

Parameter	**Values**
Number of drones	{1, 2, 3, 4, 5}
Battery endurance [min]	{30, 60}
Drone speed [mph]	{40, 60, 80}
Truck speed [mph]	35
Customer time service [min]	0.5
Drone setup time [min]	1

5.2 Configuration of the computational experiments

The algorithm and the MILP model were implemented in Python 3.9.5. The experiments were executed on the NLHPC, using 2 Intel Xeon Gold 6152 processors with 2.10 GHz, each with 22 cores and 769 GB of RAM. All experiments were run with a single thread using the CentOS Linux 7 (64-bit) operating system. Gurobi 9.1.1 is used as a general-purpose solver to solve the proposed MILP model and the reduced MILP model. Additionally, the algorithms proposed by Lu et al. (2022), the SA and GA, were modified to meet the proposed requirements of the FDTSP.

5.3 Parameter settings

The proposed MH algorithm uses a total of 11 parameters. Given the number of parameters, the algorithm's performance can be affected if the parameters are not properly adjusted. Therefore, a parameter tuning is performed using the package designed by López-Ibáñez et al. (2016) (IRACE). Ten instances selected from the *large set* are used based on previous experimentation, choosing those that showed the worst performance. The IRACE execution considered a 95% confidence level and 10,000 iterations. The calibration was performed on the NLHPC using 40 parallel threads, requiring an computing time of 125.04 hours. The tuned parameters, their description, range, and values obtained by IRACE are listed in Table 7.

Table 7: Range and values of the parameters.

Parameter	Description	Range	Final value
I^V	Number of VNS iterations.	{100000}	100000
I^S	Number of SA iterations	{10, 20, 30, 40, 50, 60, 70, 80, 90, 100}	10
K^V	Size of the VNS neighborhood.	[1, 5]	1
K^S	Size of the SA neighborhood.	[1, 5]	1
c_1	Parameter for the first procedure of VNS.	{50, 100, 150, 200, 250, 300, 350, 400, 450, 500}	150
c_2	Parameter for the second procedure of VNS.	{50, 100, 150, 200, 250, 300, 350, 400, 450, 500}	300
α	Probability associated with the use of the ICT and TRA operators.	[0, 1]	0.706
β	Temperature decrease constant of the SA.	[0.95, 1]	0.972
t_0	Initial temperature of the SA.	{10, 20, 30, 40, 50, 60, 70, 80, 90, 100, 110, 120, 130, 140, 150}	120
T	Final temperature of the SA.	$\{10^{-1}, 10^{-2}, 10^{-3}, 10^{-4}, 10^{-5}, 10^{-6}, 10^{-7}, 10^{-8}, 10^{-9}, 10^{-10}\}$	10^{-10}
P	Penalization parameter.	{500, 1000, 1500, 2000, 2500, 3000, 3500, 4000, 4500, 5000}	2000

5.4 Results of the computational experiments

In order to guarantee a fair comparison with the SA and GA proposed by Lu et al. (2022), our experiments were conducted by executing all algorithms in ten independent runs with different random seeds for each instance, considering three different time limits for the *large set* of instances. Accordingly, we defined these sets of time limits based on the sizes of the sets of the instances. Each execution was limited to 60 seconds for instances with 40 or fewer nodes, 300 seconds for instances with 60 and 80 nodes, and 600 seconds for instances with 100 and 150 nodes. This means that past the time limit, no further iteration of the algorithm is performed, but it is allowed to finish the last execution initiated before the time limit. As for the *small set* instances, the MILP model was limited to 3600 seconds and the other algorithms to 60 seconds. If the optimal solution is not found by the model, the best integer solution found is used for comparison against MH, SA, and GA.

5.4.1 Results of the *large set* of instances

This computational experiment was conducted on the *large set* of instances. We report the results obtained by the algorithms, considering both their minimum and average costs. The average cost corresponds to the average of the results obtained for an instance executed ten times, and the minimum cost correspond to the lowest cost of the ten instance runs. For each instance of the *large set*, the measure of the quality of the solutions found is the percentage difference (`gap`) between the solution found by the proposed MH, the algorithms SA and GA (Lu et al., 2022), and the best solution found by any of these algorithms. The formula for the gap is described in Equation (35), where $f(s^*)$ corresponds to the best known cost found by any of the three algorithms, and $f(s)$ is the cost found by each algorithm. Thus, if the evaluated costs correspond to the minimum costs, the gap is called $\texttt{gap}_{\texttt{min}}$. If the evaluated costs are the average costs, the gap is called $\texttt{gap}_{\texttt{avg}}$. Additionally, we consider the metric $\texttt{hit}^{\texttt{large}}$ that consists of a count of the number of times an algorithm achieves a $\texttt{gap}_{\texttt{min}}$ equal to zero for each ten independent runs.

$$\frac{f(s) - f(s^*)}{f(s^*)} \times 100\% \tag{35}$$

Tables 8 and 9 present the comparison of the results of MH, SA, and GA for the instances of the *large set*. Table 8 groups the results by averaging the costs according to the type of area and the number of nodes in the evaluated instances. Table 9 groups the results by averaging the costs according to the number of drones used to solve the instances. The first table has the following format: the first column refers to the area of the scenarios: urban, suburban, and rural. The second column indicates the number of nodes in the instance ($|V|$). Meanwhile, Table 9 starts with the first column referring to the number of drones used (D). Then, the rest of the columns follow the same format for both tables. Continuing, the next three columns are the average percentage gap values ($\texttt{gap}_{\texttt{avg}}$) of the MH, SA, and GA algorithms, respectively. Next, three columns present the minimum cost percentage gap ($\texttt{gap}_{\texttt{min}}$) for the MH, SA, and GA algorithms. Following this, the average computing times (`time`) in seconds over ten runs are shown for each algorithm, followed by the last columns corresponding to the number of hits ($\texttt{hit}^{\texttt{large}}$) achieved by each algorithm. Finally, the penultimate row shows the average gaps for each algorithm, while the last row shows the total number of hits obtained by an algorithm. It is important to note that each row in Table 8 presents the average of a metric for 30 instances that have

the same number of nodes within a specific area. As for Table 9, each row contains the metrics aggregated by the average for 54 instances solved with the same number of drones.

Moreover, we consider the three metrics proposed by Lu et al. (2022) to measure the quality of the solution for the *large set*, expressed as a percentage. The first is `Service Level`, which corresponds to the ratio between the number of customers served during 4 hours and the total number of customers. Then, `drone delivery` is the ratio of customers served by drones to the total number of customers. Finally, `waiting time` is the proportion of time the truck waits for the drones' arrival over the total travel time.

Table 8: Comparison of the algorithms by area and number of nodes.

Area	$\|V\|$	gap_{avg} (%)			gap_{min} (%)			time(s)			hit^{large}		
		MH	**SA**	**GA**	**MH**	**SA**	**GA**	**MH**	**SA**	**GA**	**MH**	**SA**	**GA**
U	**80**	3.16	11.66	17.14	0.22	7.85	10.08	302.76	300.00	303.74	25	4	1
	100	2.38	12.00	17.61	0.00	8.09	12.21	605.43	600.00	607.22	30	0	0
	150	1.44	13.35	29.04	0.00	10.82	20.36	857.27	600.00	616.74	30	0	0
S	**40**	9.03	17.78	21.32	0.85	8.96	7.78	60.00	60.00	61.09	23	2	5
	60	5.41	12.32	17.47	0.89	6.23	8.37	300.00	300.00	302.13	24	4	6
	80	5.62	12.11	16.38	1.72	7.25	9.02	302.06	300.00	303.37	20	3	7
R	**15**	11.53	27.79	22.83	0.30	16.10	14.29	60.00	60.00	60.40	26	4	4
	30	8.69	35.45	33.89	2.06	22.76	18.84	60.00	60.00	60.66	23	3	5
	40	10.08	37.76	33.31	1.36	21.29	11.95	60.00	60.00	60.95	20	4	6
Average		**6.37**	**20.02**	**23.22**	**0.82**	**12.15**	**12.55**	**289.72**	**260.00**	**264.03**	-		
Sum		-									**221**	**24**	**34**

Analyzing Table 8 shows the aggregated results according to the type of area and the number of nodes, averaging the results of the instances solved with all the quantities of drones, speeds, and battery duration considered. For the gap_{avg} metric, MH achieves the best gap for all quantities of nodes, obtaining the lowest percentage for the urban instance with 150 nodes, with 1.44%, and obtaining the best percentage difference among the algorithms for the rural case with 40 nodes, with an approximate difference of 27.68% compared to SA and 23.23% compared to GA. Reflected in the average row, MH achieves the best result with 6.37%. Furthermore, it can be seen that MH obtains, on average, better gaps as instances with a greater number of nodes are solved. When observing the results with the same number of nodes but located in different types of areas, MH tends to achieve better results in smaller areas. Regarding the gap_{min} metric, it is observed that the MH algorithm is the reference algorithm for all cases. The worst gap for MH is observed in the rural case with 30 nodes, with a 2.06% gap. However, in this case, SA exhibits a gap of 22.76%, and GA shows a gap of 18.84%. The averages of the columns reflect the behavior of MH, which consistently achieves the best results. This can be verified by

observing the columns of the $\texttt{hit}^{\texttt{large}}$ metric, where MH surpasses SA and GA in quantity for all particular cases. The total sum of hits reflects that MH, with 221, achieves the best results. Regarding computing time, the `time(s)` metric shows that MH is the algorithm requiring the largest average computing time, at 289.72 seconds. This is mainly due to the time needed to evaluate urban instances with 150 nodes, requiring approximately 250 seconds more than the other algorithms. This excess is caused by the reduced MILP model consuming the most allocated computing time due to the complexity of managing constraints and the extensive solution search process.

Table 9: Comparison of the algorithms by number of drones.

D	$\texttt{gap}_{\texttt{avg}}$ (%)			$\texttt{gap}_{\texttt{min}}$ (%)			`time(s)`			$\texttt{hit}^{\texttt{large}}$		
	MH	**SA**	**GA**	**MH**	**SA**	**GA**	**MH**	**SA**	**GA**	**MH**	**SA**	**GA**
1	4.22	20.45	30.42	0.00	12.87	19.40	301.06	260.00	268.84	54	0	0
2	5.68	20.09	23.79	0.27	12.42	12.16	292.99	260.00	264.13	48	3	5
3	6.44	20.01	21.95	0.93	11.64	12.09	283.63	260.00	262.81	42	8	5
4	7.54	20.01	20.31	1.01	12.09	10.32	284.08	260.00	262.20	41	4	10
5	7.97	19.56	19.64	1.89	11.74	8.75	286.86	260.00	262.19	36	9	14
Average	**6.37**	**20.02**	**23.22**	**0.82**	**12.15**	**12.55**	**289.72**	**260.00**	**264.03**	-		
Sum	-									**221**	**24**	**34**

Table 9 presents the aggregated results according to the number of drones used to solve the instances with different parameters. Observing the results for the metric $\texttt{gap}_{\texttt{avg}}$, the MH achieves the best values for all drone quantities. This is reflected in the average for each column, where the MH achieves the best value at 6.37%. The MH achieves the best gap, 4.22%, using one drone. It can also be noted that the gap increases as the number of drones used increases. Regarding the metric $\texttt{gap}_{\texttt{min}}$, on average, MH is the referenced algorithm for all drone quantities. Thus, it is the algorithm with the best performance, 0.82% on average. Again, the trend is observed that with a greater number of drones used, the effectiveness of the MH decreases, resulting in fewer hits. Finally, once again, the metric `time` shows that the MH requires more computing time, 289.72 seconds on average.

To assess whether the differences in the obtained results are statistically significant, we employed Wilcoxon's signed-ranks test, a non-parametric method for pairwise comparisons. This test is analogous to the Student's t-test but offers greater sensitivity and avoids the violation of assumptions required by parametric tests, such as independence, normality, and homoscedasticity, which are often not met when using metaheuristic algorithms (Carrasco et al., 2020, Derrac et al., 2011, García et al., 2009). Based on the data related to the resolution of the *large set*, the minimum costs obtained by the MH, SA, and GA are

taken. In this way, with a 95% confidence interval, each algorithm has a sample of 270 results. Thus, a p-value of 0.003 is obtained for the differences between MH and SA, and 6.28×10^{-18} between MH and GA. This indicates that for both comparisons, the null hypothesis that the median of the differences is equal to zero is rejected, verifying a statistically significant difference between the results.

When analyzing the metrics for `Drone Delivery`, `Service Level`, and `Waiting Time` proposed by Lu et al. (2022), these metrics are averaged according to the area, number of nodes, and number of drones. Accordingly, Figures 3–5 correspond to the `Drone Delivery` metric, Figures 6–8 correspond to the `Service Level` metric, and Figures 9–11 correspond to the `Waiting Time` metric. Each set of figures provides detailed insights into how these metrics vary with changes in area size, the number of nodes, and the number of drones used.

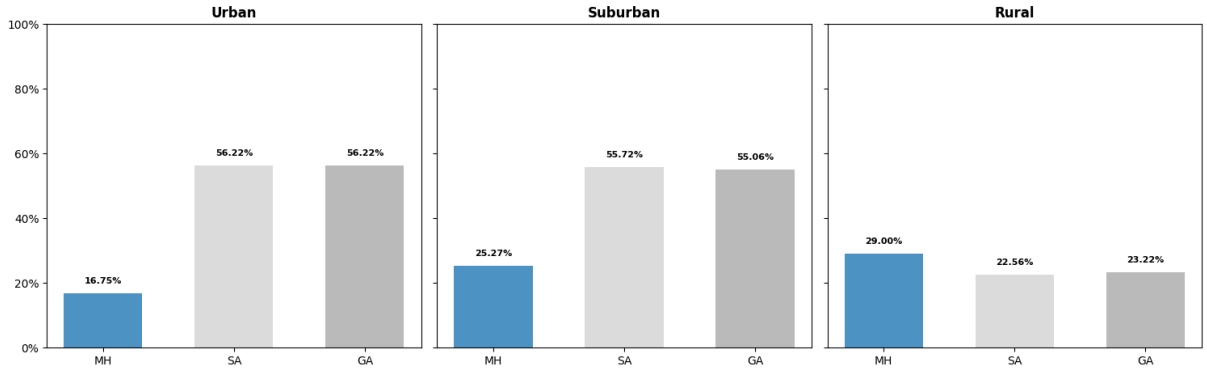

Figure 3: `Drone Delivery` grouped by Area.

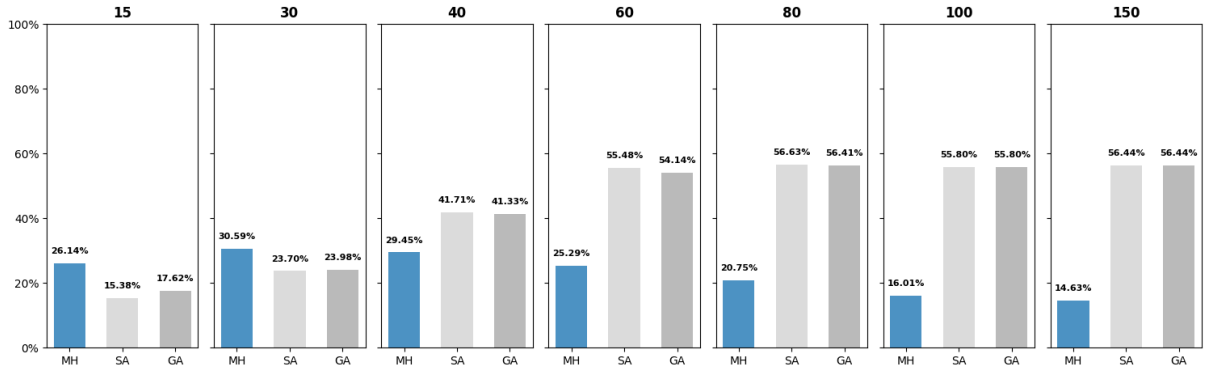

Figure 4: `Drone Delivery` grouped by # Nodes.

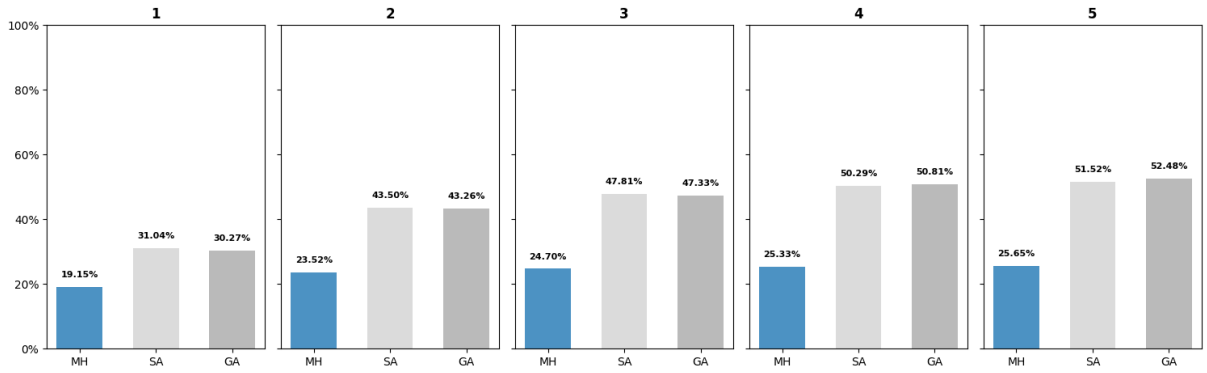

Figure 5: `Drone Delivery` grouped by # Drones.

Analyzing the `Drone Delivery` metric in Figures 3–5, we can see that the percentages obtained by the three algorithms remain constant for the different groupings, between the approximate ranges of 15% to 30% for the MH, 15% to 60% for the SA, and 17% to 60% for the GA. Overall, the SA and GA obtain a much greater drone usage percentage than the MH, and a similar usage percentage between these benchmark algorithms. It is noteworthy that MH obtains on average a higher percentage of drone usage in rural areas with a 29.00% compared to 22.56% and of the SA and 23.22% of the GA, as shown in Figure 3. With these results, it is noteworthy that increased drone usage is not essential for achieving better outcomes. Moreover, the MH algorithm achieved a better high rate of drone utilization in sprawling areas, considering a low number of nodes served.

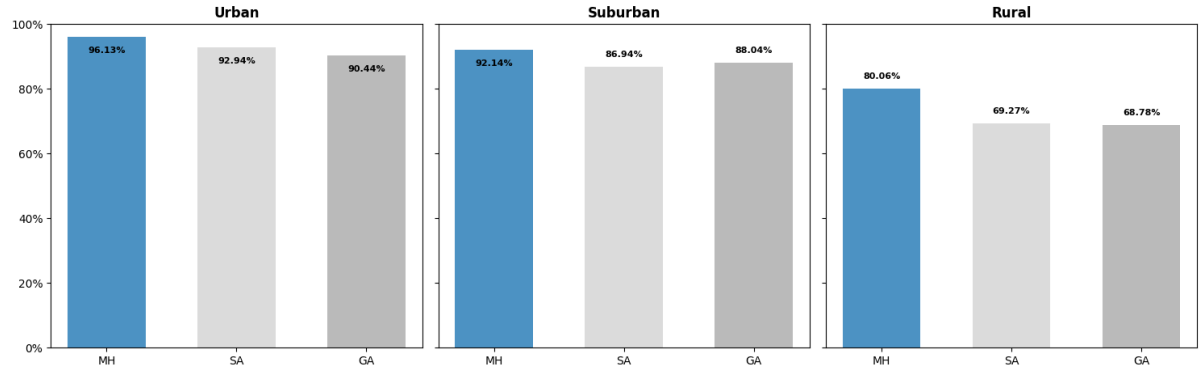

Figure 6: `Service Level` grouped by Area.

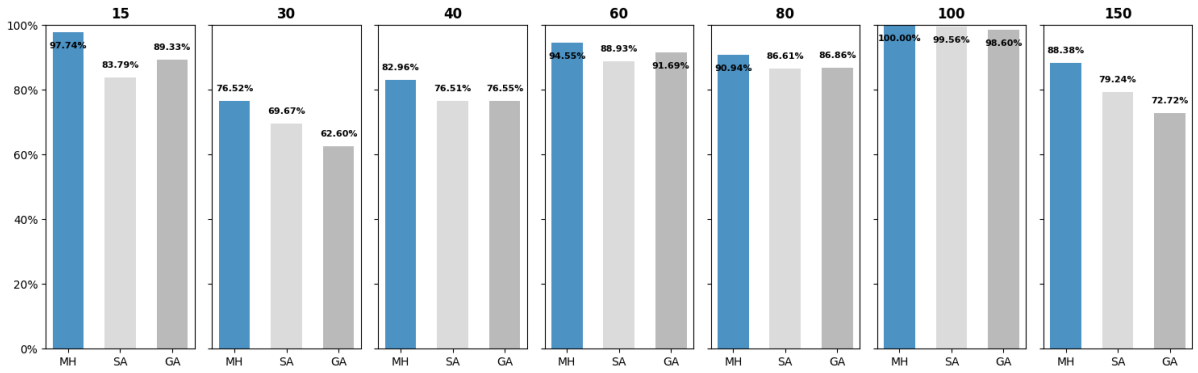

Figure 7: `Service Level` grouped by # Nodes.

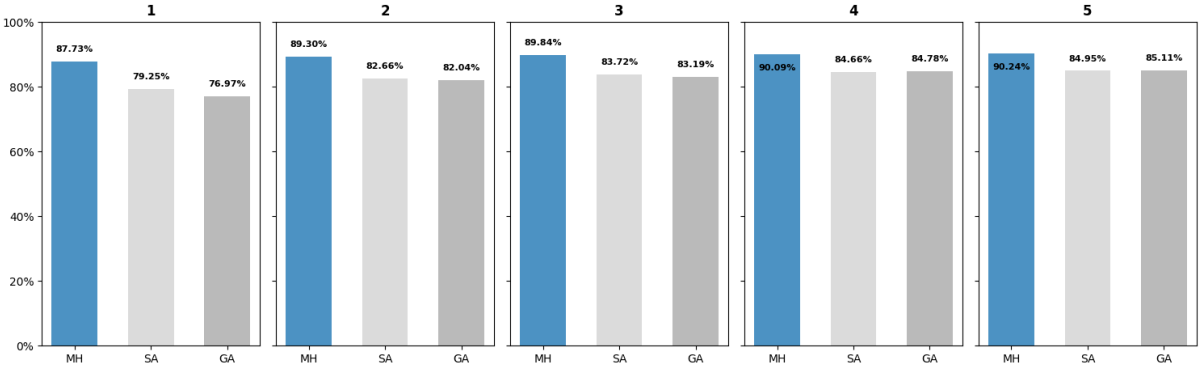

Figure 8: `Service Level` grouped by # Drones.

As for the `Service Level metric`, we can see in Figures 6–8 that all the algorithms obtain a percentage above 60% of customers served in four hours. Grouped by area, MH is the algorithm that manages to serve the highest number of customers on average, based on the percentages obtained. Then, grouping by a number of nodes, MH continues to obtain better percentages on average, obtaining in the worst case 76.52% of the attention in instances of 30 nodes, and the best case, with 100%, corresponding to instances with 100 nodes. Meanwhile, grouping according to the number of drones used, MH obtains the best attention percentages, in the worst case with an average of 87.73% for instances solved with one drone. In this way, it can be understood that the MH manages to find lower-cost solutions, which mostly have a travel time of less than four hours.

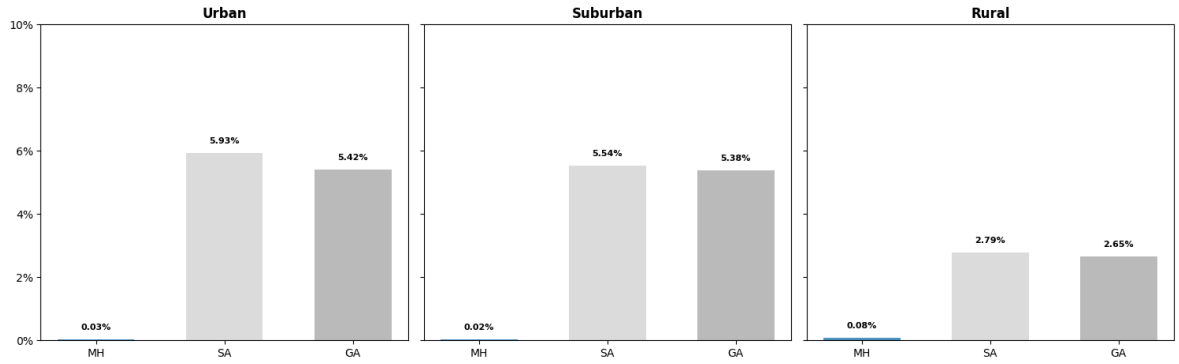

Figure 9: `Waiting Time` grouped by Area.

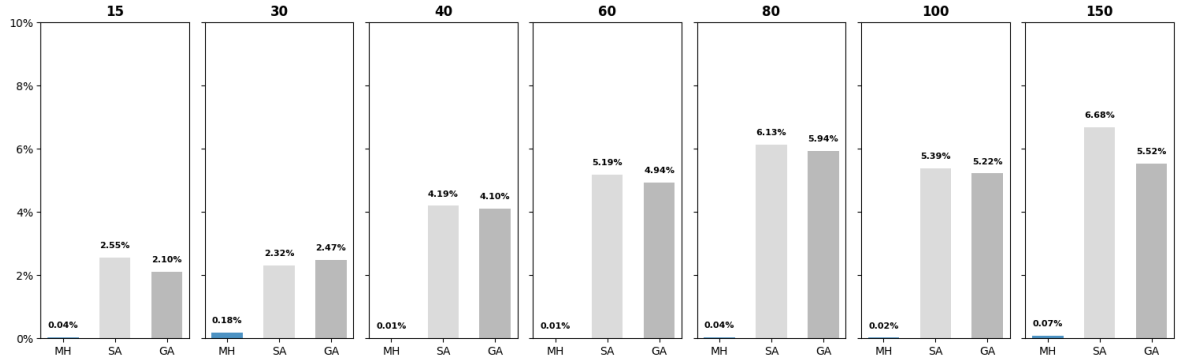

Figure 10: `Waiting Time` grouped by # Nodes.

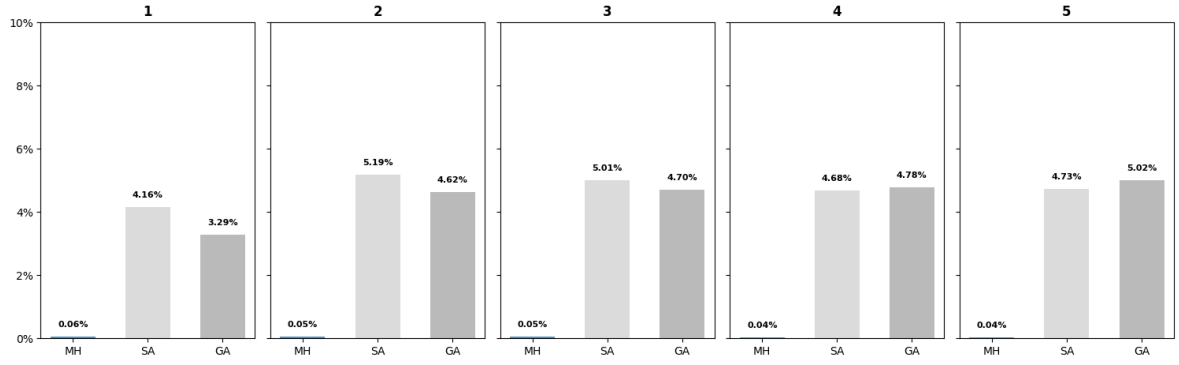

Figure 11: `Waiting Time` grouped by # Drones.

Regarding the `Waiting Time` metric, we can note from Figures 9–11 that none of the algorithms has, on average, a solution with an excessive waiting time. Remarkably, MH achieves better averages in all aggregations. In the worst case, MH has 0.08% of the time for the rural scenario in Figure 9. SA with 6.68% is the worst case, corresponding to the average for instances with 150 nodes shown in Figure 10. Meanwhile, the average for

the GA is a 5.94% grouping for 80 nodes, as shown in Figure 10. Based on these results and the `Drone Delivery` metric, it can be seen that the solutions generated by the MH algorithm are more efficient in managing drones. But, the reduced use of these UAVs also partially explains this low percentage for the metric.

5.4.2 Results of the *small set* of instances

For instances of the *small set*, the quality measure of the solutions found by the MILP model is the $\texttt{gap}_{\texttt{MILP}}$, consisting of the percentage difference between the upper bound (UB) and the lower bound (LB). This gap is shown in Equation (36).

$$\frac{UB - LB}{LB} \times 100\% \tag{36}$$

Additionally, the metric $\texttt{hit}^{\texttt{opt}}_{\texttt{MILP}}$ corresponds to the number of times the model achieves the optimal result, i.e., when the $\texttt{gap}_{\texttt{MILP}}$ is equal to zero. On the other hand, the quality measure between the solutions found by the different methods for this set of instances also corresponds to Equation (35). The difference lies in considering both the algorithms' minimum costs and the model's upper bounds. Thus, $f(s^*)$ corresponds to the best found cost, and $f(s)$ is the cost found by each method. We call the metric $\texttt{gap}_{\texttt{UB}}$. In the same way as the *large set*, the metric $\texttt{hit}^{\texttt{small}}$ is used, corresponding to the count of the number of times the model or an algorithm achieves a $\texttt{gap}_{\texttt{UB}}$ equal to zero for each run.

Tables 10 and 11 present the comparison of results for the MILP model, MH, SA, and GA for the instances of the *small set*. Table 10 is organized with the following structure for the first two columns: the first column indicates the size of the area for the types of scenarios, urban, suburban, and rural. The second column specifies the number of nodes in each instance. Meanwhile, Table 11 begins with the first column referring to the number of drones utilized to solve the instances (D). From there onward, for both tables, the next three columns correspond to the upper bound, lower bound, and the average computing time of the model. The following column reports the gap between the lower and upper bounds ($\texttt{gap}_{\texttt{MILP}}$). Then, the next four columns present the quality measure $\texttt{gap}_{\texttt{UB}}$ for the MILP model, MH, SA, and GA, respectively. The next column presents the number of $\texttt{hit}^{\texttt{opt}}_{\texttt{MILP}}$. The subsequent four columns present the number of $\texttt{hit}^{\texttt{small}}$ obtained by the MILP, MH, SA, and GA. Finally, the penultimate row shows the average of the gap columns calculated for the model and each algorithm. The last row shows the total

number of hits obtained, with each method having a maximum of 270 due to the number of instances resolved considering a given number of drones.

Table 10: Comparison of MILP model and the algorithms by area and number of nodes.

Area	$\|V\|$	UB	LB	time(s)	$\texttt{gap}_{\texttt{MILP}}$ (%)	$\texttt{gap}_{\texttt{UB}}$ (%)				$\texttt{hit}^{\texttt{opt}}_{\texttt{MILP}}$	$\texttt{hit}^{\texttt{small}}$			
						MILP	MH	SA	GA		MILP	MH	SA	GA
U	**10**	28.03	27.32	1479.85	1.77	0.00	0.00	80.21	78.72	29	30	30	0	0
	15	52.73	4.33	3600.08	91.30	1.94	0.00	46.97	43.82	0	24	30	0	0
	20	65.92	2.90	3600.31	95.50	9.69	0.50	14.84	14.32	0	14	26	0	4
S	**10**	64.74	53.78	1962.05	13.53	0.00	0.00	53.50	51.68	23	30	30	0	0
	15	82.24	6.40	3600.08	91.27	0.00	0.00	79.25	87.65	0	30	30	0	0
	20	136.13	6.41	3600.30	95.20	19.20	3.08	10.11	12.60	0	11	22	8	7
R	**10**	176.04	159.62	2047.22	9.53	0.00	0.00	53.68	47.33	26	30	30	0	0
	15	216.95	33.56	3600.09	82.63	0.00	0.00	42.88	42.67	0	30	30	0	0
	20	267.35	19.80	3600.36	92.33	7.73	0.29	22.44	21.37	0	13	28	1	1
Average		**121.12**	**34.90**	**3010.04**	**63.67**	**4.28**	**0.43**	**44.88**	**44.46**		**-**			
Sum		-								**78**	**212**	**256**	**9**	**12**

Considering the aggregated results by type of area and number of nodes for all drone quantities and parameters. Table 10 shows that, on average, the MILP model does not obtain an upper bound equal to the lower bound. If we observe the column for the $\texttt{gap}_{\texttt{MILP}}$ metric, we see that the instances with ten nodes achieve the best results on average, having the best gap for urban instances with 1.77%. In fact, by observing the $\texttt{hit}^{\texttt{opt}}_{\texttt{MILP}}$ metric, it is possible to notice that the model achieves optimality 78 times in total, and all these solutions are found in instances with ten nodes. It can also be observed that as the number of nodes evaluated increases, the gap increases considerably. Regarding computing times, only for the instances with ten nodes is less than an hour of computation required on average. Observing the results for the $\texttt{gap}_{\texttt{UB}}$ metric, it is evident that the MILP model and the MH are the referenced methods for comparison. MH achieves a 0.43% and the MILP model 4.28% on average. Considering that the MILP model reaches optimality for some instances, and MH achieves a $\texttt{gap}_{\texttt{UB}}$ equal to zero for the same instances, it can be said that the MH also achieves the optimal result for those instances. The $\texttt{hit}^{\texttt{small}}$ metric shows that both methods achieve the majority of hits. However, the MH obtains the highest number with 256. This difference is because the fact that the MH gets more hits in instances with a higher number of nodes, where the solver used for the MILP model requires an average of one hour to obtain only feasible solutions, resulting in worse outcomes compared to the MH.

Table 11: Comparison of MILP model and the algorithms by number of drones.

D	UB	LB	time(s)	$\texttt{gap}_{\texttt{MILP}}$ (%)	$\texttt{gap}_{\texttt{UB}}$ (%)				$\texttt{hit}^{\texttt{opt}}_{\texttt{MILP}}$	$\texttt{hit}^{\texttt{small}}$			
					MILP	MH	SA	GA		MILP	MH	SA	GA
1	148.19	40.07	3328.12	67.26	10.66	0.00	26.95	25.30	13	34	54	0	0
2	127.08	36.48	3128.13	64.91	5.12	0.95	32.50	34.26	15	40	50	4	4
3	121.00	37.68	3026.74	61.15	3.88	0.88	35.89	35.77	18	43	49	3	5
4	106.17	30.79	2784.18	62.44	0.82	0.00	59.11	58.70	16	48	54	0	0
5	103.18	29.49	2783.01	62.61	0.94	0.32	69.95	68.28	16	47	49	2	3
Average	**121.12**	**34.90**	**3010.04**	**63.67**	**4.28**	**0.43**	**44.88**	**44.46**	-				
Sum	-								**78**	**212**	**256**	**9**	**12**

Regarding the aggregating results by the number of drones used, we can again show that the MILP model does not achieve an upper bound equal to the lower bound on average. In the $\texttt{gap}_{\texttt{MILP}}$ metric, a trend can be observed where solving instances using three drones can achieve better average results compared to using one or two. However, when using four or five drones, the gap worsens. Meanwhile, the average computing time required, as the number of drones increases, the computing time decreases. In terms of optimality, the $\texttt{hit}^{\texttt{opt}}_{\texttt{MILP}}$ shows that the MILP model achieves the highest number of optimal solutions when using more than one drone. The $\texttt{hit}^{\texttt{small}}$ metric reflects that the MILP model and the MH achieve the majority of hits, but the MH obtains the highest number with 256 in total and also the highest number for all quantities of drones used.

Chapter 6

Conclusions

This thesis proposes a MILP model and a matheuristic algorithm to address the FDTSP. A literature review was conducted focusing on the solution methods of various authors. The matheuristic is composed of the combined use of a reduced MILP model and a VNS. This metaheuristic iteratively searches for solutions through two procedures: one focused on alternating the service of truck to customers, and the other focused on improving the route of the drones. Additionally, several instances with different complexities were evaluated, differentiated by the area covered, the number of nodes, drone capacities, and the number of these UAVs.

Computational results provide key insights, revealing that the matheuristic consistently obtains better results on average across all instances, and it achieves the lowest cost in most cases regardless of the number of drones used. The Wilcoxon test indicates that the results of the metaheuristic are statistically different from those of the algorithms in the literature. In terms of computing time, the proposed algorithm manages to obtain solutions within the established time limit. However, it significantly exceeds this limit on average for instances with 150 nodes, partly due to the use of the reduced MILP model. The efficiency of the matheuristic increases if a larger number of nodes in a small area are evaluated. Besides, better results are obtained on average if fewer drones are considered to solve the problem. In terms of the metrics related to the operation outlined in the solutions, it is evident that the quality of a solution is not necessarily linked to a high percentage of drone usage. Additionally, it can be observed that shorter route times are achieved, most within a four-hour window, without significantly delaying the

truck. Regarding evaluating the MILP model, it is possible to solve small-scale problems with up to 10 nodes to optimality in less than an hour using the Gurobi Optimizer. The matheuristic also achieves optimal results in various instances with ten nodes.

Further research could address several aspects of both proposals. One possible improvement for the algorithm is to achieve greater efficiency related to using of a larger number of drones. To accomplish this, a possible strategy is to intensify the local search and diversification operators so that a greater number of routes for the UAV are evaluated in one iteration. On the side of the proposed MILP model, it is evident that the number of nodes in the problem is a limitation. Therefore, a solution to this could be proposing a new exact algorithm based on $B\&C$ or a similar method. Additionally, efforts can be made to apply the problem modeling to real operational conditions between a truck and drones. In some cases, this would involve modifying the assumptions of the problem due to the complexity of a real operation. Therefore, modifications to the proposed model and algorithm would also required.

References

Agatz, N., Bouman, P., & Schmidt, M. (2018). Optimization approaches for the traveling salesman problem with drone. *Transportation Science*, 52(4):965–981.

AlMuhaideb, S., Alhussan, T., Alamri, S., Altwaijry, Y., Aljarbou, L., & Alrayes, H. (2021). Optimization of truck-drone parcel delivery using metaheuristics. *Applied Sciences*, 11(14).

Amazon (2024). Amazon drones can now fly farther and deliver to more customers following faa approval. *https://www.aboutamazon.com/news/transportation/amazon-drone-prime-air-expanded-delivery-faa-approval*. Accessed: June 08, 2024.

Bouman, P., Agatz, N., & Schmidt, M. (2018). Dynamic programming approaches for the traveling salesman problem with drone. *Networks*, 72(4):528–542.

Cai, M. & Qian, H. (2023). A hybrid variable neighborhood search algorithm on routing and scheduling for truck-assisted multi-drone parcel delivery. In *2023 IEEE 6th Information Technology, Networking, Electronic and Automation Control Conference (ITNEC)*, volume 6, pages 1725–1729.

Carrasco, J., García, S., Rueda, M., Das, S., & Herrera, F. (2020). Recent trends in the use of statistical tests for comparing swarm and evolutionary computing algorithms: Practical guidelines and a critical review. *Swarm and Evolutionary Computation*, 54:100665.

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

Chang, Y. S. & Lee, H. J. (2018). Optimal delivery routing with wider drone-delivery areas along a shorter truck-route. *Expert Systems with Applications*, 104:307–317.

Croes, G. A. (1958). A method for solving traveling-salesman problems. *Operations Research*, 6(6):791–812.

Dell'Amico, M., Montemanni, R., & Novellani, S. (2022). Exact models for the flying sidekick traveling salesman problem. *International Transactions in Operational Research*, 29(3):1360–1393.

Dell'Amico, M., Montemanni, R., & Novellani, S. (2021). Algorithms based on branch and bound for the flying sidekick traveling salesman problem. *Omega*, 104:102493.

Derrac, J., García, S., Molina, D., & Herrera, F. (2011). A practical tutorial on the use of nonparametric statistical tests as a methodology for comparing evolutionary and swarm intelligence algorithms. *Swarm and Evolutionary Computation*, 1(1):3–18.

DJI (2024). Dji completes world's first drone delivery tests on mount everest. *https://www.dji.com/global/media-center/announcements/dji-completes-world-first-drone-delivery-tests-on-mount-everest-en*. Accessed: June 08, 2024.

El-Adle, A. M., Ghoniem, A., & Haouari, M. (2023). A variable neighborhood search for parcel delivery by vehicle with drone cycles. *Computers and Operations Research*, 159:106319.

Faithfull, M. (2024). Walmart drone dream has been cleared for take off in dallas. *https://www.forbes.com/sites/markfaithfull/2024/05/13/walmart-drone-dream-has-been-cleared-for-take-off-in-dallas/*. Accessed: June 08, 2024.

Ferrandez, S. M., Harbison, T., Weber, T., Sturges, R., & Rich, R. (2016). Optimization of a truck-drone in tandem delivery network using k-means and genetic algorithm. *Journal of Industrial Engineering and Management*, 9(2):374–388.

García, S., Molina, D., Lozano, M., & Herrera, F. (2009). A study on the use of non-parametric tests for analyzing the evolutionary algorithms' behaviour: a case study on the CEC'2005 Special Session on Real Parameter Optimization. *Journal of Heuristic*, 15(6):617–644.

Garg, V., Niranjan, S., Prybutok, V., Pohlen, T., & Gligor, D. (2023). Drones in last-mile delivery: A systematic review on efficiency, accessibility, and sustainability. *Transportation Research Part D: Transport and Environment*, 123:103831.

Ha, Q. M., Deville, Y., Pham, Q. D., & Hà, M. H. (2018). On the min-cost traveling salesman problem with drone. *Transportation Research Part C: Emerging Technologies*, 86:597–621.

Helsgaun, K. (2000). An effective implementation of the lin–kernighan traveling salesman heuristic. *European Journal of Operational Research*, 126(1):106–130.

Kirkpatrick, S., Gelatt, C. D., & Vecchi, M. P. (1983). Optimization by simulated annealing. *Science*, 220(4598):671–680.

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

Lu, S.-H., Kuo, R., Ho, Y.-T., & Nguyen, A.-T. (2022). Improving the efficiency of last-mile delivery with the flexible drones traveling salesman problem. *Expert Systems with Applications*, 209:118351.

López-Ibáñez, M., Dubois-Lacoste, J., Pérez Cáceres, L., Birattari, M., & Stützle, T. (2016). The irace package: Iterated racing for automatic algorithm configuration. *Operations Research Perspectives*, 3:43–58.

Madani, B. & Ndiaye, M. (2022). Hybrid truck-drone delivery systems: A systematic literature review. *IEEE Access*, 10:92854–92878.

Mahmoudi, B. & Eshghi, K. (2022). Energy-constrained multi-visit tsp with multiple drones considering non-customer rendezvous locations. *Expert Systems with Applications*, 210:118479.

Metropolis, N., Rosenbluth, A. W., Rosenbluth, M. N., Teller, A. H., & Teller, E. (1953). Equation of State Calculations by Fast Computing Machines. *The Journal of Chemical Physics*, 21(6):1087–1092.

Mladenović, N. & Hansen, P. (1997). Variable neighborhood search. *Computers and Operations Research*, 24(11):1097–1100.

Mohd Daud, S. M. S., Mohd Yusof, M. Y. P., Heo, C. C., Khoo, L. S., Chainchel Singh, M. K., Mahmood, M. S., & Nawawi, H. (2022). Applications of drone in disaster management: A scoping review. *Science & Justice*, 62(1):30–42.

Murray, C. C. & 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.

Nguyen, M. A., Sano, K., & Tran, V. T. (2020). A monte carlo tree search for traveling salesman problem with drone. *Asian Transport Studies*, 6:100028.

Pope, C. (2024). Drone deliveries pass ice-cream test with flying colours in dublin trial. *https://www.irishtimes.com/food/2024/06/08/drone-deliveries-pass-ice-cream-test-with-flying-colours-in-dublin-trial/*. Accessed: June 08, 2024.

Rejeb, A., Abdollahi, A., Rejeb, K., & Treiblmaier, H. (2022). Drones in agriculture: A review and bibliometric analysis. *Computers and Electronics in Agriculture*, 198:107017.

Rinaldi, M., Primatesta, S., Bugaj, M., Rostáš, J., & Guglieri, G. (2023). Development of heuristic approaches for last-mile delivery tsp with a truck and multiple drones. *Drones*, 7(7).

Roberti, R. & Ruthmair, M. (2021). Exact methods for the traveling salesman problem with drone. *Transportation Science*, 55(2):315–335.

Roberts, N. B., Ager, E., Leith, T., Lott, I., Mason-Maready, M., Nix, T., Gottula, A., Hunt, N., & Brent, C. (2023). Current summary of the evidence in drone-based emergency medical services care. *Resuscitation Plus*, 13:100347.

Rogers, K. (2024). Drone startup zipline hits 1 million deliveries, looks to restaurants as it continues to grow. *https://www.cnbc.com/2024/04/19/autonomous-drone-startup-zipline-hits-1-million-deliveries.html*. Accessed: June 08, 2024.

Salama, M. R. & 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.

Schaumann, S. K., Kundu, A., Pina-Pardo, J. C., Winkenbach, M., Gatica, R. A., Wagner, S. M., & Matis, T. I. (2024). The flying sidekick traveling salesman problem with multiple drops: A simple and effective heuristic approach. arXiv.

Schermer, D., Moeini, M., & Wendt, O. (2020). A branch-and-cut approach and alternative formulations for the traveling salesman problem with drone. *Networks*, 76(2):164–186.

Tiniç, G. O., Karasan, O. E., Kara, B. Y., Campbell, J. F., & Ozel, A. (2023). Exact solution approaches for the minimum total cost traveling salesman problem with multiple drones. *Transportation Research Part B: Methodological*, 168:81–123.

Tirkolaee, E. B., Cakmak, E., & Karadayi-Usta, S. (2024). Traveling salesman problem with drone and bicycle: multimodal last-mile e-mobility. *International Transactions in Operational Research*.

Tu, P. A., Dat, N. T., & Dung, P. Q. (2018). Traveling salesman problem with multiple drones. In *Proceedings of the 9th International Symposium on Information and Communication Technology*, SoICT '18, page 46–53, New York, NY, USA. Association for Computing Machinery.

Vásquez, S. A., Angulo, G., & Klapp, M. A. (2021). An exact solution method for the tsp with drone based on decomposition. *Computers and Operations Research*, 127:105127.

Yılmaz, C., Cengiz, E., & Kahraman, H. T. (2024). A new evolutionary optimization algorithm with hybrid guidance mechanism for truck-multi drone delivery system. *Expert Systems with Applications*, 245:123115.

Zhang, J. & Woensel, T. V. (2023). Dynamic vehicle routing with random requests: A literature review. *International Journal of Production Economics*, 256:108751.

Zhang, Q., Chan, F. T., Chung, S.-H., & Fu, X. (2024). A matheuristic for aircraft maintenance routing problem incorporating cruise speed control. *Expert Systems with Applications*, 242:122711.